Allen-Bradley

Bulletin 1771 Communication Controller Module

(Cat. No. 1771–KE, KF)

User Manual

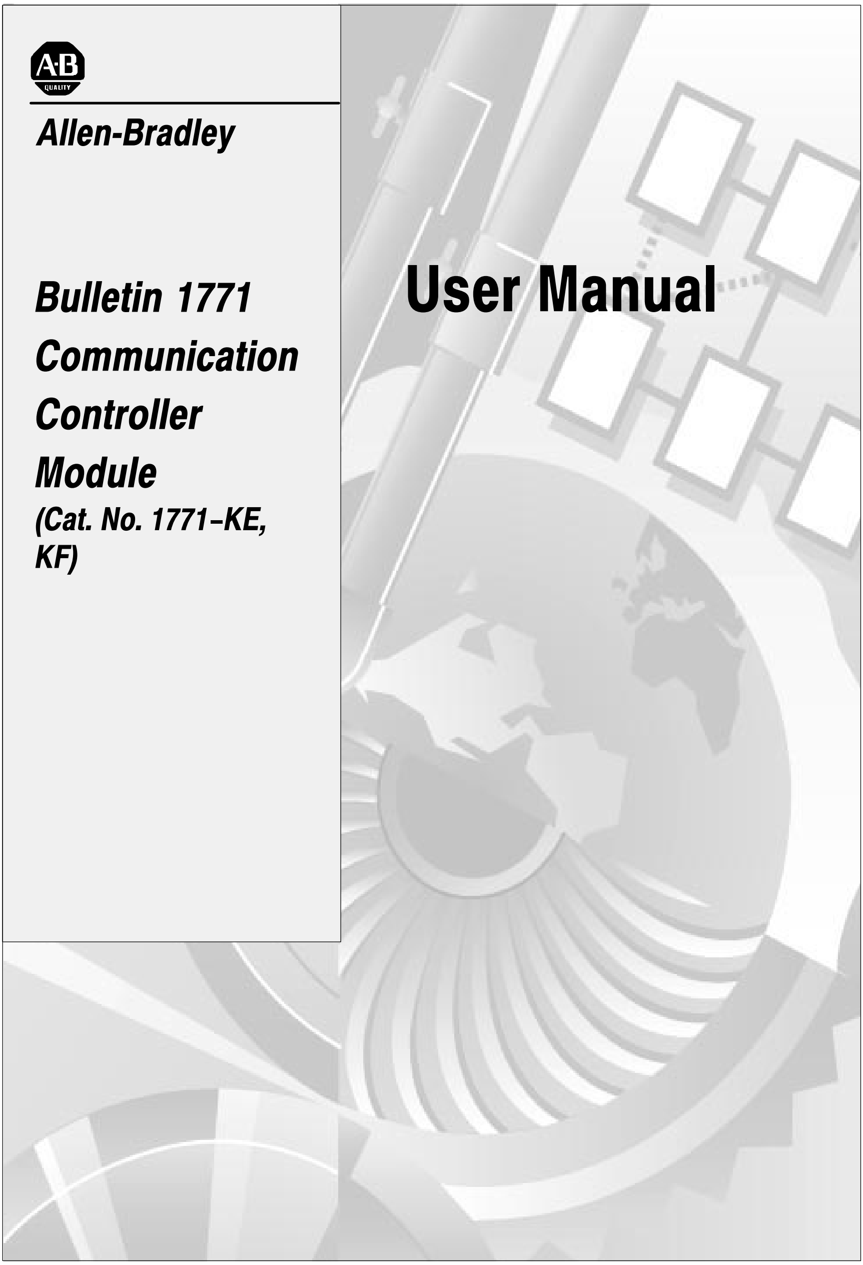


Table of Contents

Chapter 1

Introduction

General

Communication Controller Modules (cat. nos. 1771-KE and 1771-KF) link intelligent RS-232-C devices to an Allen-Bradley Data Highway. Each of these modules gives you a choice of two protocols on its RS-232-C link:

- Full duplex
- Half duplex

The 1771-KE and 1771-KF modules perform the same functions. They differ only in the way they are mounted and in the way they receive power. Where these differences are discussed, each module is referenced separately. Otherwise, both modules are referred to collectively as the KE/KF module.

About This Manual

This manual describes installation, operation, and communication protocols of the KE/KF module, and it assumes that you are already thoroughly familiar with how to program your computer or other intelligent RS-232-C device. It does not assume prior knowledge of the Allen-Bradley Data Highway.

Table 1.A lists related Data Highway documentation that might be helpful in conjunction with this manual. For more details about the programming and operation of specific Allen-Bradley programmable controllers, refer to the appropriate user’s manual for that controller.

This manual is organized as follows:

- Chapter 2 - Explains some Data Highway concepts
- Chapter 3 - Tells how to install a KE/KF module
- Chapters 4, 5, and 6 - Describe the communication protocol use by a KE/KF module
- Chapter 7 - Summarizes error reporting

Module Description

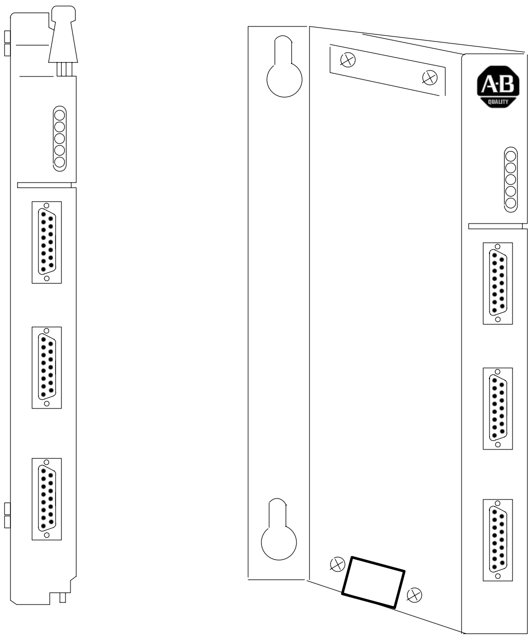
Figure 1.1 shows both the 1771-KE and 1771-KF modules. These modules have the following hardware features:

- Diagnostic indicators
- Connectors for Data Highway and RS-232-C devices
- Communication option switches
- Stand-alone mounting bracket (1771-KF only)
- Terminal strip for power supply connection (1771-KF only)

Table 1.A
Related Data Highway Documentation

Publication Number	Title
1770-810	Data Highway Cable Assemble and Installation Manual
1770-843	Network Communication Software User's Manual
1770-801	Communication Adapter Module (cat. no. 1771-KA) User's Manual
1771-802	Communication Controller Module (cat. no. 1771-KC/KD) User's Manual
1771-811	PLC-2 Family/RS-232-C Interface Module (cat. no. 1771-KG) User's Manual
1773-801	PLC-4 Communication Interface Module (cat. no.1773-KA) User's Guide
1774-819	Communication Adapter Module (cat. no. 1774-KA) User's Manual
1775-802	PLC-3 Communication Adapter Module (cat. no. 1775-KA) User's Manual

Figure 1.1
Communication Controller Modules



Cat. No. 1771-KE

Cat. No. 1771-KF

Specifications

Table 1.B lists the specifications for a KE/KF module.

Table 1.B
KE/KF Communication Controller Module Specifications

Communication Rates	**Cabling**
Data highway: 57,600 bits per second (recommended) RS-232-C: switch-selectable from 110 to 19,200 bits per second	Data highway: Data highway dropline cable (cat. no. 1770-CD) RS-232-C: Data terminal Interface Cable (cat. no.1770-CG) or Modem Interface Cable (cat. no. 1770-CP)
Functions	**Power Requirements**
Interface a programmable RS-232-C compatible device with an Allen-Bradley Data Highway Serve as a replacement for 1771-KC/KD Communication Controller Modules	1.2A @ 5V DC **Power Source** 1771-KE: Bulletin 1771 I/O rack power supply 1771-KF: User supplied (Allen-Bradley cat. no. 1771-P2 or similar)
Location	**Ambient Temperature Rating**
1771-KE: single slot in Bulletin 1771 I/O rack 1771-KF: stand-alone mounting	0^{0}C to 60^{0}C (32^{0}F to 140^{0}F) operational -40^{0}C to 85^{0}C(-40^{0}F to 185^{0}F) storage
Communication Ports	**Ambient Humidity Rating**
Data Highway RS-232-C	5% to 95% without condensation

Applications

A KE/KF module provides either point-to-point link or a multidrop link between an Allen-Bradley Data Highway and an intelligent RS-232-C device. By “intelligent RS-232-C device” we mean any device that complies with RS-232-C electrical standards and that can be programmed to handle the communication protocol described in chapters 4 through 6 of this manual. Throughout this manual, we will also use the term “computer” in a general sense to refer to this type of device. Some examples include:

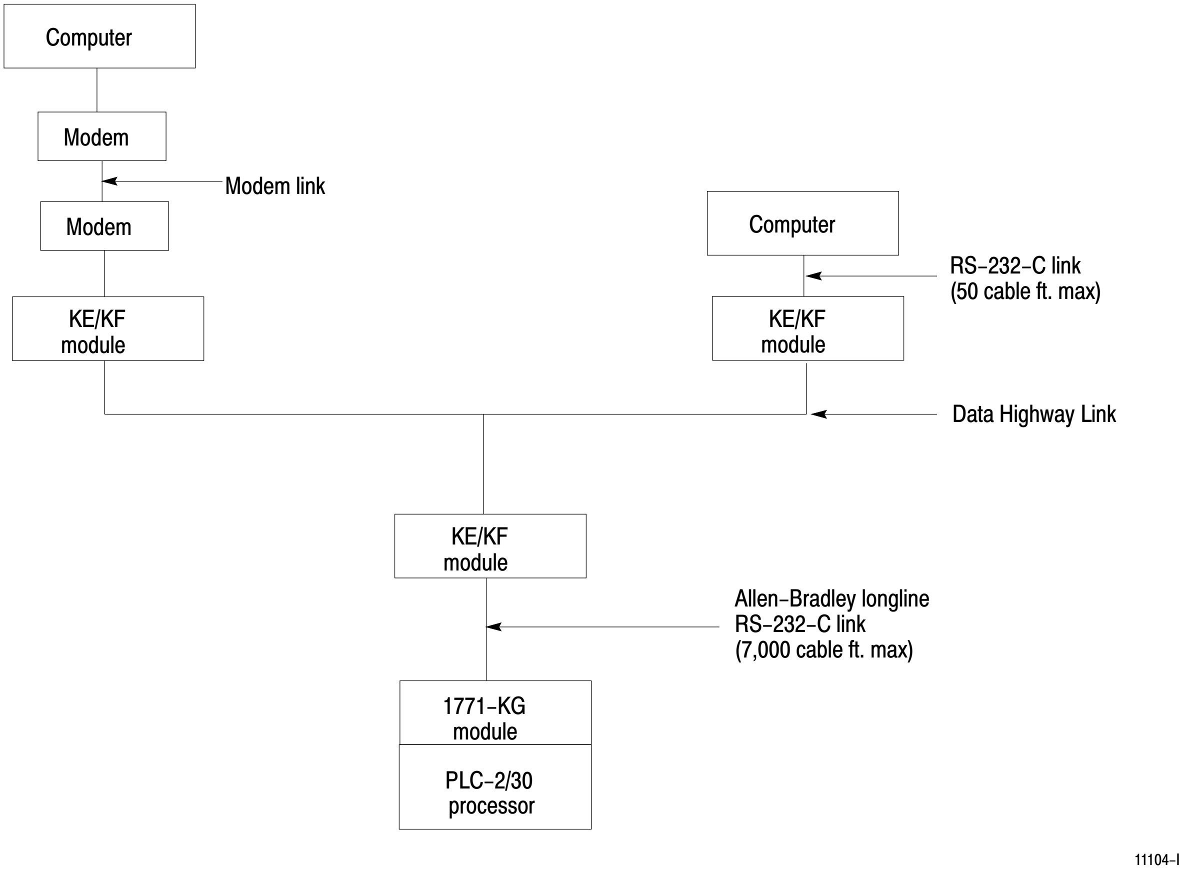
- An Allen-Bradley Advisor™ Color Graphics System
- A PLC-3 Programmable Controller and connected Communication Adapter Module (cat. no. 1775-KA)
- A PLC-2 Family Programmable Controller and connected PLC-2 Family/RS-232-C Interface Module (cat. no. 1771-KG)

- A PLC-4 Microtrol Programmable Controller and connected Communication Interface Module (cat. no. 1773-KA)
- A variety of minicomputers and microcomputers

In point-to-point configuration, the KE/KF module connects one intelligent RS-232-C device as a single station on a Data Highway. Figure 1.2 illustrates this configuration. Point-to-point links can use either peer-to-peer (full duplex) or master-slave (half duplex) communication.

In a multidrop configuration, one intelligent RS-232-C device connects to several Data Highways through sets of modems and KE/KF modules. Figure 1.3 illustrates this type of configuration. If the multidrop link consists of broadband modems, you can select either peer-to-peer (full duplex) or master-slave (half duplex) communication. If the multidrop link consists of baseband modems, you must use master-slave (half duplex) communication because baseband modems support only one communication channel.

Figure 1.2
Point-to-Point Links

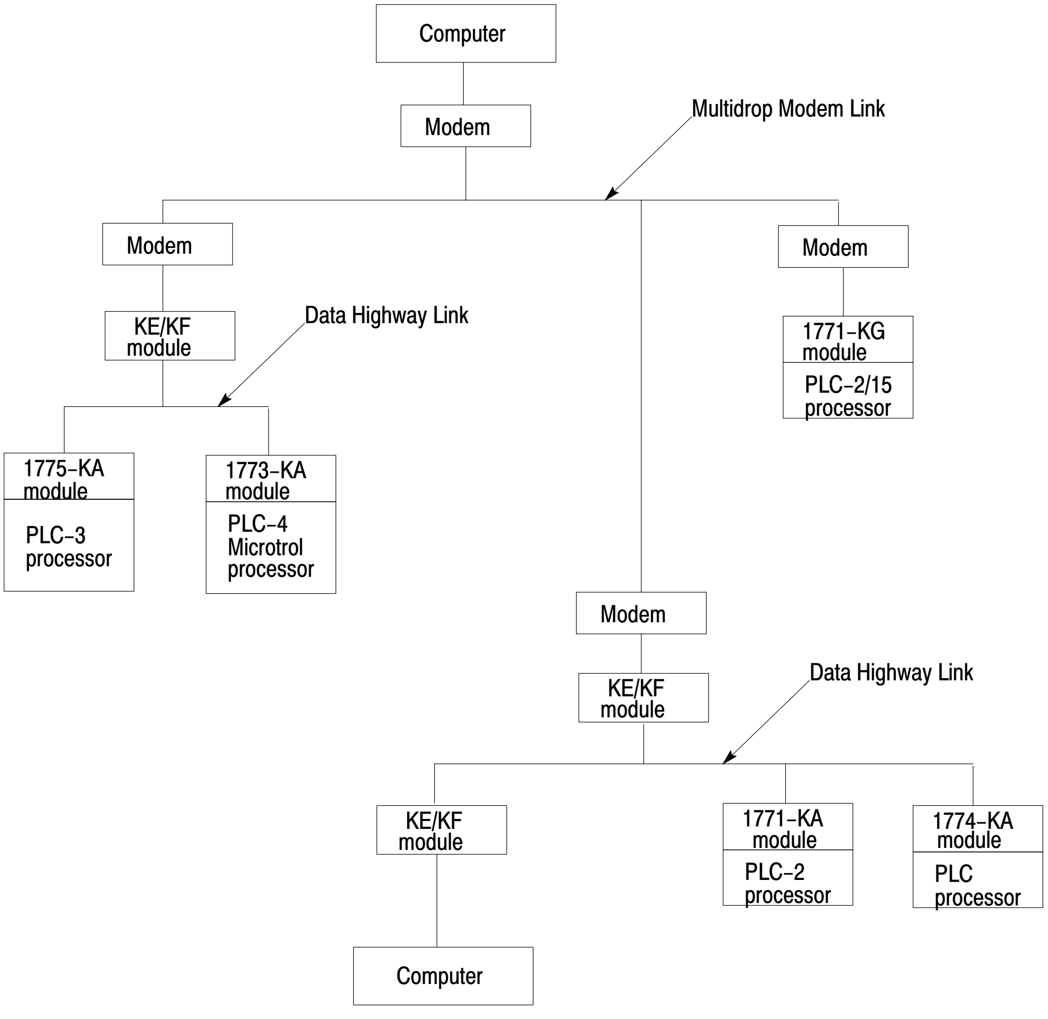


In either type of configuration, there are three possible ways you can connect the KE/KF module:

- Direct connection to an intelligent RS-232-C device if the KE/KF module is mounted within 50 cable feet of the device
- Longline connection to an Allen-Bradley 1775-KA, 1773-KA, or 1771-KG interface module if the KE/KF module is within 7,000 cable feet of the other module
- Modem connection if the KE/KF module is within 50 cable feet of an RS-232-C compatible modem

You may also use the 1771-KE module to replace a 1771-KC module, or the 1771-KF module to replace a 1771-KD module, in an existing application. By properly setting some option switches on the KE/KF module, you can make this replacement without having to change any application programs that you were using with the 1771-KC/KD module. Refer to section titled "RS-232-C Communication Rate, Parity, and Diagnostic Commands" in chapter 3 for an explanation of how to set the KE/KF option switches.

Figure 1.3
Multidrop Link



Chapter 2

Communication Concepts

General

This chapter presents some of the concepts of communication with the KE/KF module. It describes the physical communication links to the module and the various levels of software necessary to make those links work.

A KE/KF module connects a computer or programmable controller to an Allen-Bradley Data Highway. In doing so, the module acts as an interface between two physical communication links:

- Data Highway link
- RS-232-C link

The Data Highway link provides peer-to-peer communication between the module and other stations on the Data Highway. It uses a half-duplex (polled) protocol and rotation of link mastership.

In addition to a physical link layer, communication on either the Data Highway or the RS-232-C link involves three levels of software programming:

- Application layer
- Network management layer
- Data link layer

If you are using a computer on a RS-232-C link, you must program all three layers. For the Data Highway, you need program only the application layer; the Data Highway interface modules automatically take care of the other two layers.

The rest of this chapter presents some of the concepts behind the physical communication links and their three software layers. For more details on the application and network layers, refer to chapters 5 and 6. For more information on the data link layer of the RS-232 link, refer to chapter 4.

Physical Link Layer

The physical link layer is a set of cables and interface modules that work together to provide a channel for communication between the various points, called stations, on the physical link. A station consists of an intelligent programmable device (e.g., PC or computer) and the module or modules that interface it with the physical link.

In this way, the KE/KF module allows stations on one link to communicate with stations on the other link. Since these two physical links have different communication protocols, the KE/KF module serves mainly as a protocol translator.

Data Highway

The Data Highway is a local area network (LAN) that can allow peer-to-peer communication among 64 stations. Figure 2.1 illustrates a Data Highway network.

The Data Highway link consists of a trunkline that can be up to 10,0900 feet long and droplines that can be up to 100 feet each. Each station is at the end of a dropline.

The Data Highway link implements peer-to-peer communication through a modified token-passing scheme called the floating master. With this arrangement, each station has equal access to become the master. The stations bid for temporary mastership based on their need to send information.

Unlike a master/slave relationship, a floating master relationship does not require the current master to poll each station to grant permission to transmit. Therefore, it provides a more efficient network because there is less overhead per transaction.

Stations

A station consists of a computer or PC processor and the module or modules that interface it with the Data Highway link. Within a station that contains a KE/KF module, an RS-232-C link is required as an auxiliary link to the Data Highway. Figure 2.1 shows three such stations.

One station consists of an Advisor Color Graphic System connected to a KE/KF module through an RS-232-C link limited to 50 cable-ft.

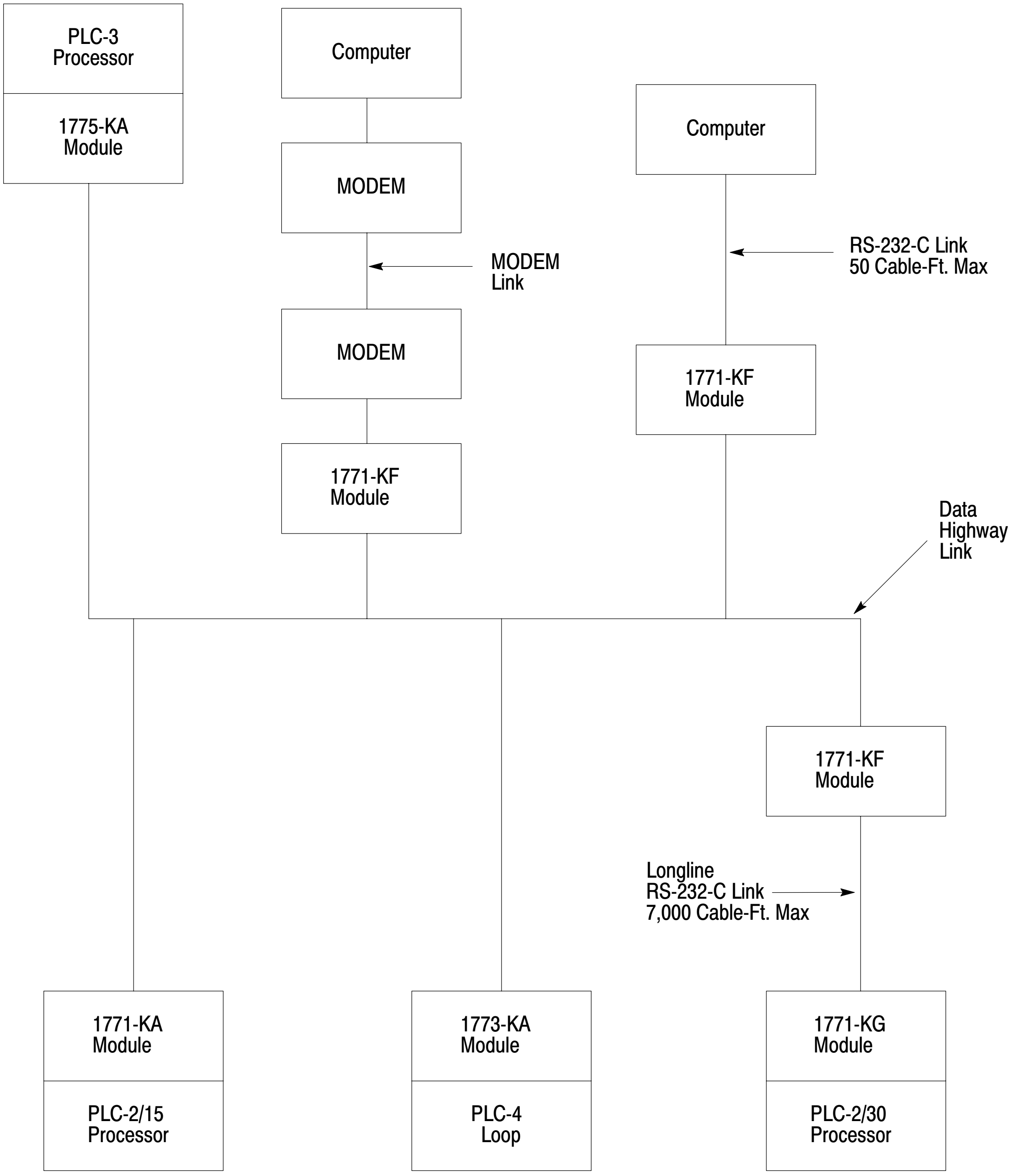
Another station consists of a computer interfacing with a KE/KF module through modem link that is limited only by the nature of the modems themselves.

The third such station consists of a 1773-KA module interfacing a PLC-4 Microtrol loop with a KE/KF module through longline RS-232-C link limited to 7,000 cable-ft. If you want a link longer than 7,000 ft., you must use modems.

PC Programming

All Allen-Bradley PC processor can connect to the Data Highway through an appropriate station interface module. All of these processors can receive and reply to command messages, and some of them can also transmit command messages. For an explanation of how to program PCs to send and receive messages, refer to the user's manual for the appropriate station interface module.

Figure 2.1
Data Highway Network



11106–I

Computer Programming

The communication protocol for the Data Highway link is transparent to a computer on the network. However, for a computer to send or receive messages through the Data Highway network, it must be programmed to communicate with its KE/KF module over an RS-232-C link. Chapters 4, 5, and 6 describe the protocol that you must program your computer to use on this RS-232-C link.

Stand-Alone Links

A stand-alone communication link is totally separate from any Data Highway network. Through use of interface modules other than the KE/KF, your computer can communicate directly with one or more PCs over an RS-232-C link. Two PCs can also communicate directly over a similar RS-232-C link.

PC-to-PC

Figure 2.2 shows two possible stand-alone PC-to-PC communication links. Each is a point-to-point link in which two PC processors can communicate as peers. Ladder diagram programs in the PC processors initiate the transfer of messages between stations.

Figure 2.2
Stand-alone PC to PC Links

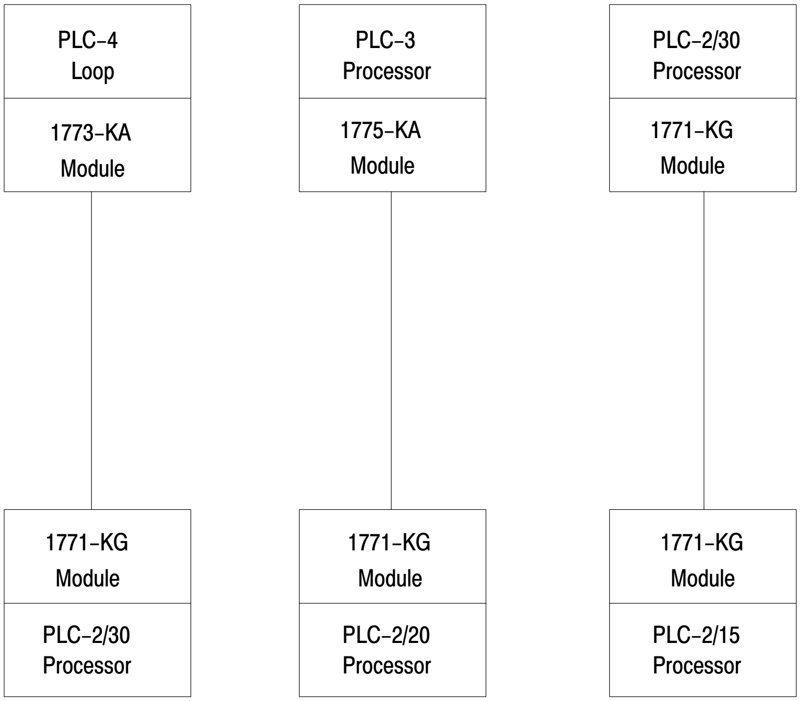
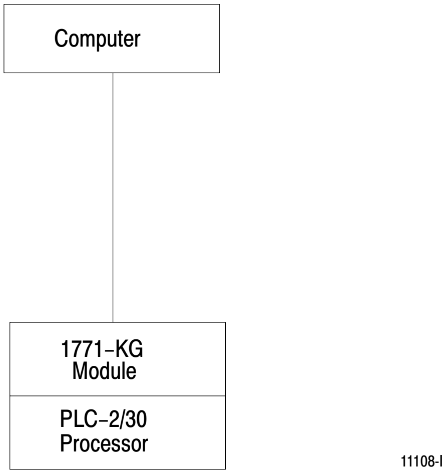


Figure 2.3
Stand-alone Point-to-Point Link to a Computer



Both of the PC-to-PC links in Figure 2.2 are longline RS-232-C links limited to 7,000 cable feet each. If you need a longer distance, you can use modems to create such a link.

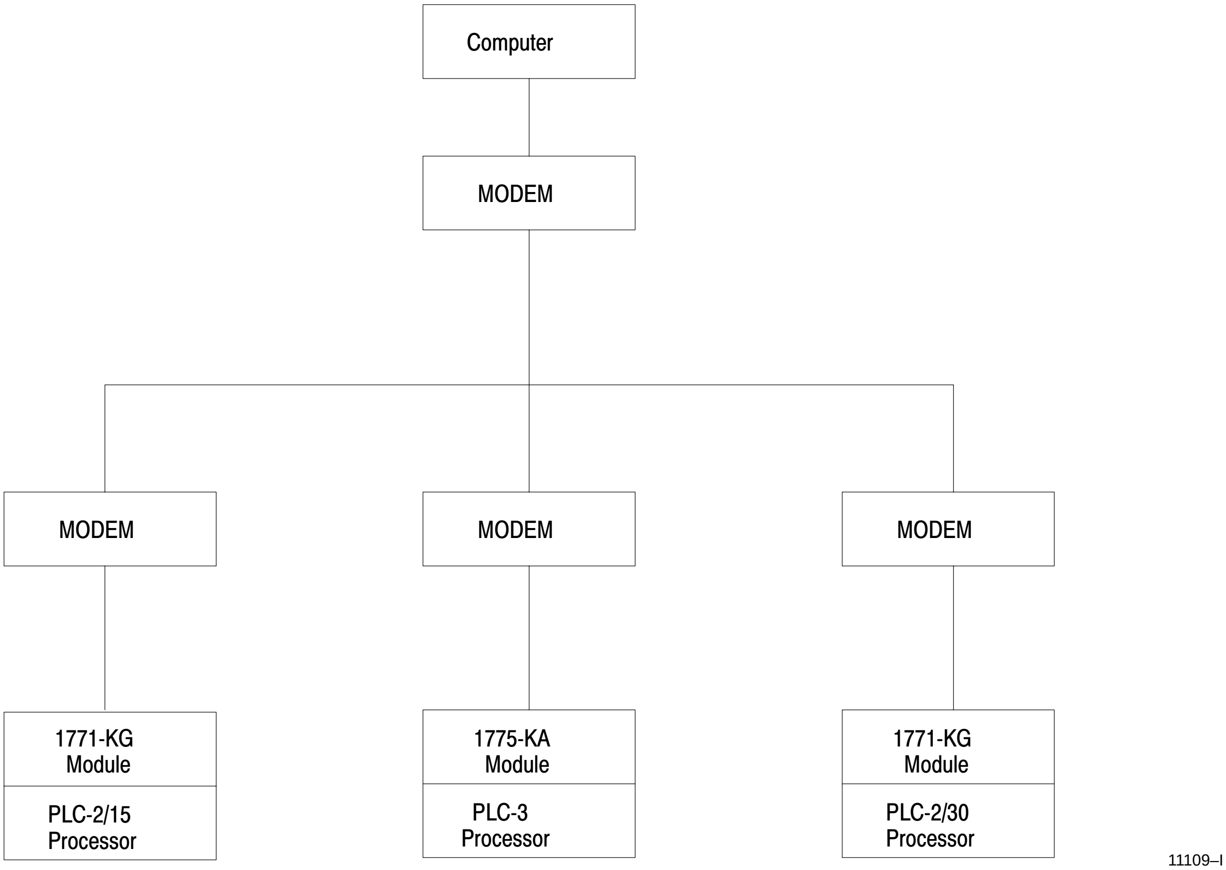
Computer-to-PC

A computer can communicate directly with PC through either a point-to-point or a multidrop RS-232-C link.

Figure 2.3 shows a point-to-point link to a computer. This is an RS-232-C link limited to 50 cable feet. If you need a longer distance, you can use modems to create this link.

Figure 2.4 shows multidrop link between a computer and three PC stations. The computer can communicate with each of the PC stations directly. This type of configuration requires a modem link.

Figure 2.4
Stand-Alone Multidrop Link to a Computer



For a point-to-point link, or a multidrop broadband modem link, you can use either peer-to-peer or master-slave communication protocol. For a multidrop baseband modem link, you must use a master-slave communication protocol because the link can support only one communication channel.

A computer can send or receive messages through a stand-alone link in the same way as through a Data Highway network. To do this, the computer must be programmed to follow the communication protocol described in chapters 4, 5, and 6.

Configuration Considerations

Allen-Bradley manufactures a variety of communication interface modules for different applications. At present, these modules are:

- PLC Computer Interface Module (cat. no. 1772-CI)
- PLC Communication Adapter Module (cat. no. 1774-KA)
- PLC-2-Family Communication Adapter Module (cat. no. 1771-KA)
- PLC-3 Communication Adapter Module (cat. no. 1775-KA)

- PLC-4 Communication Interface Module (cat. no. 1773-KA)
- PLC-2-Family/RS-232-C Interface Module (cat. no. 1771-KG)
- Communication Controller Module (cat. no. 1771-KC, -KD,-KE,KF)

The following sections summarize the uses of these modules.

PC-Processor/Data Highway Interface

The following modules provide an interface between a PC processor and a Data Highway communication link:

- PLC Communication Adapter Module (cat. no. 1774-KA)
- PLC-2-Family Communication Adapter Module (cat. no. 1771-KA)
- PLC-3 Communication Adapter Module (cat. no. 1775-KA)
- PLC-4 Communication Interface Module (cat. no. 1773-KA)

PC-Processor/RS-232-C Interface

The following modules provide an interface between a PC processor and an RS-232-C communication link:

- PLC-3 Communication Adapter Module (cat. no. 1775-KA)
- PLC-4 Communication Interface Module (cat. no. 1773-KA)
- PLC-2-Family/RS-232-C Interface Module (cat. no. 1771-KG)

RS-232-C/Data Highway Interface

The following modules provide an interface between an RS-232-C communication link and a Data Highway communication link:

- Communication Controller Module (1771-KC,-KD,-KE,-KF)

The 1771-KC and 1771-KE modules must be installed in an I/O chassis. The 1771- KD and 1771-KF modules are stand-alone modules.

The 1771-KC and 1771-KD modules can provide peer-to-peer communication only through an RS-232-C link that cannot connect to a modem link. They are superseded by the 1771-KE and 1771-KF modules that can provide either peer-to-peer or master-slave communication through an RS-232-C link, which you can optionally connect to a modem link.

Configuration Selection

Figure 2.1 through Figure 2.4 illustrate several configurations in which PC stations can communicate with each other and with computers through Data Highway ports and RS-232-C ports on the station interface modules. Each configuration is useful, depending on your application.

If you want to provide a peer-to-peer communication between two or more PCs and/or a computer, use a Data Highway network as shown in Figure 2.1. For distances longer than the Data Highway itself provides, you can use an auxiliary longline RS-232-C link or modem link.

With two stations, you may want a stand-alone link. For a stand-alone link, a modem link can be used to provide communication between stations more than 10,000 cable feet apart. Also, with the full-duplex (peer-to-peer) protocol and embedded responses, it could be faster than a Data Highway link because it wouldn't be burdened with polling.

A Data Highway link has a communication rate of 57,600 bits per second and a half-duplex (peer-to-peer, polled) protocol. An RS-232-C link has a selectable communication rate up to 19,200 bits per second and a selectable protocol of half-duplex (master-slave, polled) or full-duplex (peer-to-peer, unpolled).

For a stand-alone link to a computer, you may want to use the peer-to-peer communication protocol for maximum speed. Or you may want to use the master-slave communication protocol so that the computer can select the times it will communicate over the link.

A master-slave communication protocol can be selected for any link to a computer. A peer-to-peer communication protocol can be selected only for a point-to-point link or a broadband modem multidrop link to a computer.

Even with only two stations, you may want a Data Highway link. The Data Highway provides the flexibility of easy re-configuration or expansion if you want to be able to add more stations later, and it also provides more error checking than an RS-232-C link.

Software Layers

Each of the physical links just described requires three layers of software to enable communication to take place. The layers are defined as follows:

- Application layer - controls and executes the actual tasks, or commands, specified in the communication between stations. To program this layer, use the commands described in chapter 5.
- Network management layer - handles queuing, sequencing, routing, and error status reporting for communication. If your physical link contains only Allen-Bradley PCs, you do not have to program this layer. Otherwise, refer to chapter 5 for a description of how to program this layer for an RS-232-C link to a computer.
- Data link layer - controls the flow of communication over the physical link by establishing, maintaining, and releasing the communication channel between stations. If your physical link contains only Allen-Bradley PCs, you do not have to program this layer. Otherwise, refer to chapter 4 for a description of how to program this layer for an RS-232-C link to a computer.

Application Layer

The application layer concerns the specific commands that you can program at a given station to cause that station to communicate over the link. This layer is the same for both RS-232-C and Data Highway links.

The types of commands that a station can transmit and receive vary with the type of processor at that station. Chapter 5 describes the commands that each type of PC processor can transmit or receive. To program your computer to communicate with a PC, use the appropriate command message formats shown in chapter 5.

Message Structures

All messages on a Data Highway network have the same fundamental structure, regardless of their function or destination. If you could freeze a block while it is in transmission, you would see two types of message bytes:

- Protocol bytes
- Data bytes

The methods by which these bytes are filled is determined by the nature of the station from which the transmission block originates. For example, if a transaction originates from a PC station, the station interface module automatically fills the protocol bytes. If the transaction originates from a computer station, your computer software must supply the necessary protocol. In both cases, the data bytes contain information supplied by application programs.

Command/Reply Cycle

Any transaction on a Data Highway network consists of two parts:

- A command
- A reply

This provides extra data integrity by ensuring that a required action always returns some sort of status, whether an error code or data. As a frame of reference, the command initiator is always referred to as a local station, and reply initiator is always referred to as a remote station. Unless noted otherwise, whether in a Data Highway link or an RS-232-C link, the discussion will be limited to a single local station and a single remote station.

The network layer protocol distinguishes a command from a reply. Obviously, the data area of a command and its corresponding reply depends on the type of command.

Priority

Each message on a Data Highway link is classified as either:

- High priority
- Normal priority

The priority levels of messages determine the order in which stations are polled and allowed to transmit messages. In the polling process, stations with high priority messages will always be given priority over stations with normal priority messages.

You specify the priority level for each command message. The command code contains this specification. The station that receives a command message must establish the same priority level for its corresponding reply message.

NOTE: Stations with high priority messages are given priority over stations with normal priority messages throughout the command/reply message cycle. For this reason, a command should be given a high priority designation only when special handling of specific data is required. Using an excessive number of high priority commands defeats the purpose of this feature and could delay or inhibit the transmission of normal priority messages.

Command Structures

There are four basic types of command on a Data Highway network or a stand-alone link:

- Read
- Write
- Diagnostic
- Mode select

Reads

There are two types of read:

- Physical
- Unprotected

A physical read allows you to read any area of PC memory at a remote station. However, a PC processor cannot originate a physical read command; only a computer can originate a physical read.

An unprotected read can access only the data table area of PC memory. Both computers and PCs can initiate unprotected reads.

Any read can request up to 122 words of contiguous data from PC memory.

Writes

We can classify write commands both by their application and by their level of memory access.

As an application issue, writes are divided between bit writes and word writes. Bit writes allow the local station to control bits in the data table of a remote station.

Word writes allow the local station to write up to 121 contiguous words of data into the remote station's memory, provided you abide by the restriction on memory access, discussed next.

As with reads, writes also are classified by the level of access to PC memory. Non-physical writes can access only the data table at a remote PC; physical writes can access all of user memory, including PC program memory.

Non-physical writes can be further subdivided into protected and unprotected. Protected writes can access only specified areas of the remote PC's data table memory. The accessible areas are defined by memory protection rungs in the remote PC's ladder diagram program. Unprotected writes, on the other hand, can access any area of the remote PC's data table.

In most cases, switch settings on the remote station's interface module can disable the module from executing each of these types of write commands.

Diagnostics

Diagnostic commands have to originate from a device other than a PC. You can use these commands to return status information from a remote or local station or to alter some onboard parameters at a station interface module. Diagnostic commands are particularly useful during a startup or during on-line debugging.

Mode Select

Mode select commands allow you to load a new PC program from a remote computer station. The operation of these commands varies by PC processor type. These commands can be issued only by a computer.

Network Management Layer

The network management layer is concerned with the specifics of conveying a message safely from its source to its destination. This layer is the same for both RS-232-C and Data Highway links.

If your physical link contains only Allen-Bradley PCs, you do not have to program anything for this layer; the communication interface modules automatically take care of it. If your physical link contains a computer, then refer to chapter 5 for a description of how to program this layer at the computer station.

The rest of this section explains the network management layer for the Data Highway. For the most part, you need not be concerned with the interaction of station interface modules on the Data Highway. This means that your application programs at the PCs and computers along the Data Highway need not involve themselves with inter-station protocol, handshaking, or control of the Data Highway link. This is all carried out automatically by the station interface modules. However, an understanding of station interaction is useful both to computer programmers and PC programmers. It allows optimized use of Data Highway commands and fault diagnostics.

Error Checking

Error codes can be generated at two places: remote station modules and local station interface modules. For codes that are returned from a local station module, two types of conditions can exist:

- Application programs use the wrong message format or issue illegal commands.
- The local station cannot complete a transaction due to network problems.

A remote station can return only the codes associated with an application problem at the remote station. Typically, these involve either the PC processor being off-line (in Program mode, for example) or the command trying to access memory areas blocked by either the interface module or the user application program.

In the network layer protocol, command message status is returned in a reply status byte. A value of zero in the status byte indicates successful transmission of the corresponding command. It is up to the local application program to display and act on the value of the returned byte.

Data Link Layer

The data link layer controls the flow of communication on the physical link by acquiring and releasing access to the communication channel for each station. This layer differs for each type of physical link.

Chapter 4 explains how to program the data link layer for an RS-232-C link. The rest of this section describes the data link layer of the Data Highway.

Note that you do not have to program the data link layer for the Data Highway; the communication interface modules automatically take care of it. The description of it presented here is solely for information purposes.

The protocol that is used between stations on the Data Highway link is a modified low-level implementation of HDLC that features bit stuffing, flag definition, and generation of the cyclic redundancy check (CRC).

Floating Master

Central to the interaction of Data Highway modules is the concept of the floating master. With this arrangement, no single station is permanent master controlling the Data Highway communication link at all times. Instead, each station bids for mastership, based on its need to send command or reply messages. This arrangement has two major features:

- Multiple masters
- Peer-to-peer communication

One advantage of floating master operation is that no single station disables communication on the Data Highway as long as other stations continue to operate. This means that even with disconnection or faulted operation of a module or a processor, communication between other operating station interface modules continues. This minimizes the need for backup in some applications.

When a station gains control of the Data Highway to transmit messages, it has become a master station. All other station interface modules assume a slave mode. This enables these stations to receive and acknowledge messages sent to them. Basically, a station has three states of operation:

- Transmitting messages
- Polling to determine which station gets mastership next
- Receiving messages and polls

Thus, each Data Highway station can transmit and receive both messages and polling sequences.

Figure 2.5 shows the change of states at a Data Highway station.

Message Transmission

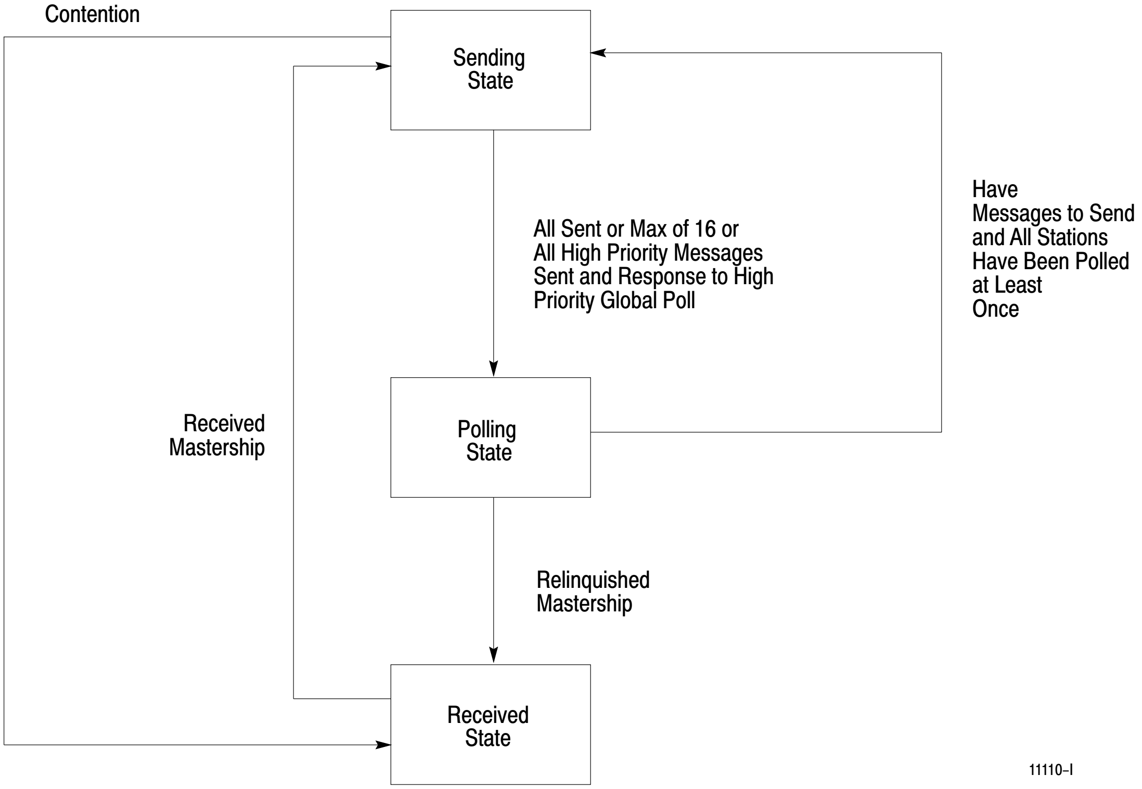
A station must have mastership of the Data Highway before it can transmit any messages or polling sequences. As part of the data integrity of the highway, all commands must receive a reply before a transaction is considered complete. Since the highway treats commands and replies as the same type of message, it takes at least one change of mastership to complete a single transaction.

Any command has to be formatted in the application program of the local, or transmitting, station. For a PC, the format is part of the PC user program. For a computer, the formatting has to be done as part of the computer program (chapter 5).

A reply message is generated by a station in response to a command message it receives. The reply message indicates that the command message was received and that the station interface module has completed the sequence of events required of it for command execution. For commands that read data, the reply message contains the data specified by the command. For commands that write data, the reply message indicates that the write operation has been completed at the receiving station.

When the replying station is a programmable controller, the reply message is an automatic function of the interface module operation and is transparent to your program. If the replying station is a computer, you must program the computer to formulate the response and the reply message.

Figure 2.5
State Transition at Master Station


Contention
Sending State
All Sent or Max of 16 or All High Priority Messages Sent and Response to High Priority Global Poll
Have Messages to Send and All Stations Have Been Polled at Least Once
Received Mastership
Polling State
Relinquished Mastership
Received State
11110–I

Polling

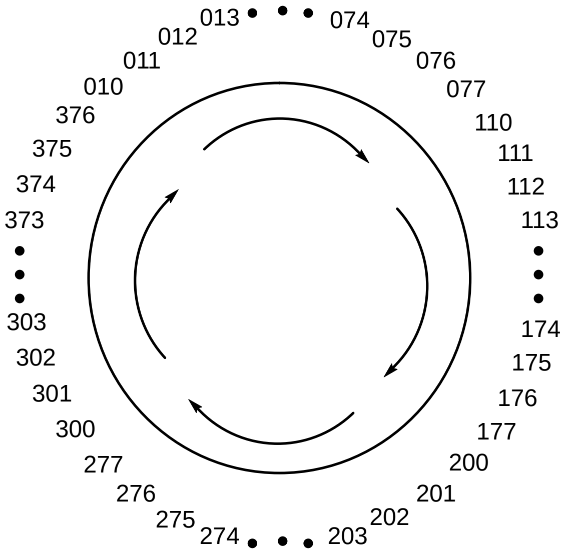
To transfer mastership between stations, the station interface modules use an orderly polling scheme. Normally, mastership passes from one station to another in a round robin fashion. During installation, you assign each station a unique number between 001 and 376 (octal). Each master passes mastership to the station with the next higher station number that is requesting mastership. All polling arithmetic is performed modulo-256 relative to the station number of the current master. For example, when the master is 200 (octal), station 220 will receive mastership before station 177, since 220 is closer in sequence to 200 than is 177. Figure 2.6 illustrates this polling scheme.

The mechanics of the polling algorithm are essentially transparent to you. This is because the communication modules handle this automatically. However, the polling algorithm does lead to the following basic rules that you should follow to optimize your Data Highway performance:

- Number your stations sequentially whenever possible.
- Keep the number of high priority messages as low as possible.

Large numbers of high priority messages slow all traffic on the network. In general, you should limit the number of high priority messages to less than 1% of the total traffic on the Data Highway.

Figure 2.6
Polling Scheme



11111–I

Data Security

There are two checks used in Data Highway network message transmission:

- A 16-bit CRC (cyclic redundancy check) on a Data Highway link
- A 8-bit BCC (block check character) or a 16 bit CRC (series A/rev. H only) on an RS-232-C link

Some station interface modules also have a switch that lets you select a parity check (even parity only) on an RS-232-C link.

A block check is used to detect errors at the link level. Beginning with revision H, you can select a CRC instead of this block check. Any device connected to the RS-232-C link must be capable of generating a BCC.

A BCC is added to the end of every message block to help detect errors of transmission between station interface modules. The command station generates a BCC by first summing every byte of the text (excluding control characters), then taking the 2’s complement of that sum. The result is the BCC. Any final carry-out bit is ignored in the BCC computations.

The receiving station also sums the text bytes, then adds that sum to the BCC to produce zero. Any sum other than zero indicates an error has been made in the transmission and causes the receiving station to respond with DEL NAK.

The CRC is used at the data link layer to validate messages transmitted on the Data Highway link.

The station interface module at the sending station appends the CRC to the message. The CRC is based on the bit pattern of the transmitted message. The receiving station also computes a CRC based on the received data and checks this against the CRC value included with the received message. A discrepancy between the transmitted CRC and the CRC computed by the receiving station indicates some fault in the transmission. If the received and computed CRC values do not agree, the message is not accepted as valid.

Link Disconnect

Floating master operation continues normally as long as all stations hare mastership of the Data Highway link. However, if any one station retained continuous control of the communication link due to a fault condition, floating master operation would not be possible and Data Highway communication would be disabled. As a preventive measure against this type of situation, each station interface module has automatic link disconnect circuitry. If a module detects that it is not relinquishing mastership of the Data Highway, this circuitry can disconnect the module from the Data Highway link. The other stations on the Data Highway can then continue to function normally.

Chapter 3

Installation

General

This chapter explains how to install the 1771-KE and 1771-KF modules. There are five parts to installation:

- Setting the communication option switches
- Mounting the module
- Connecting the module to its power supply
- Connecting the module to the Data Highway and RS-232-C links
- Observing the diagnostic indicators

Please read the first three chapters of this manual carefully before attempting any of the installation steps.

Communication Option Switches

The KE/KF module has 6 switch assemblies (Figure 3.1) that enable you to select various communication options. The switch assemblies and their corresponding options are:

Switch Assembly	Communication Option
SW-1	RS-232-C link features
SW-2, SW-3, SW-4	Station number
SW-5	Data Highway communication rate
SW-6	RS-232-C link communication rate and parity

If you have a revision A-G module:

Read section titled “RS-232-C Link Features Revisions A-G” to learn how to set your switches in switch assembly SW-1.

If you have a revision H module:

Read section titled “RS-232-C Link Features Revision H” to learn how to set your switches in switch assembly SW-1.

Figure 3.1
Communication Option Switches

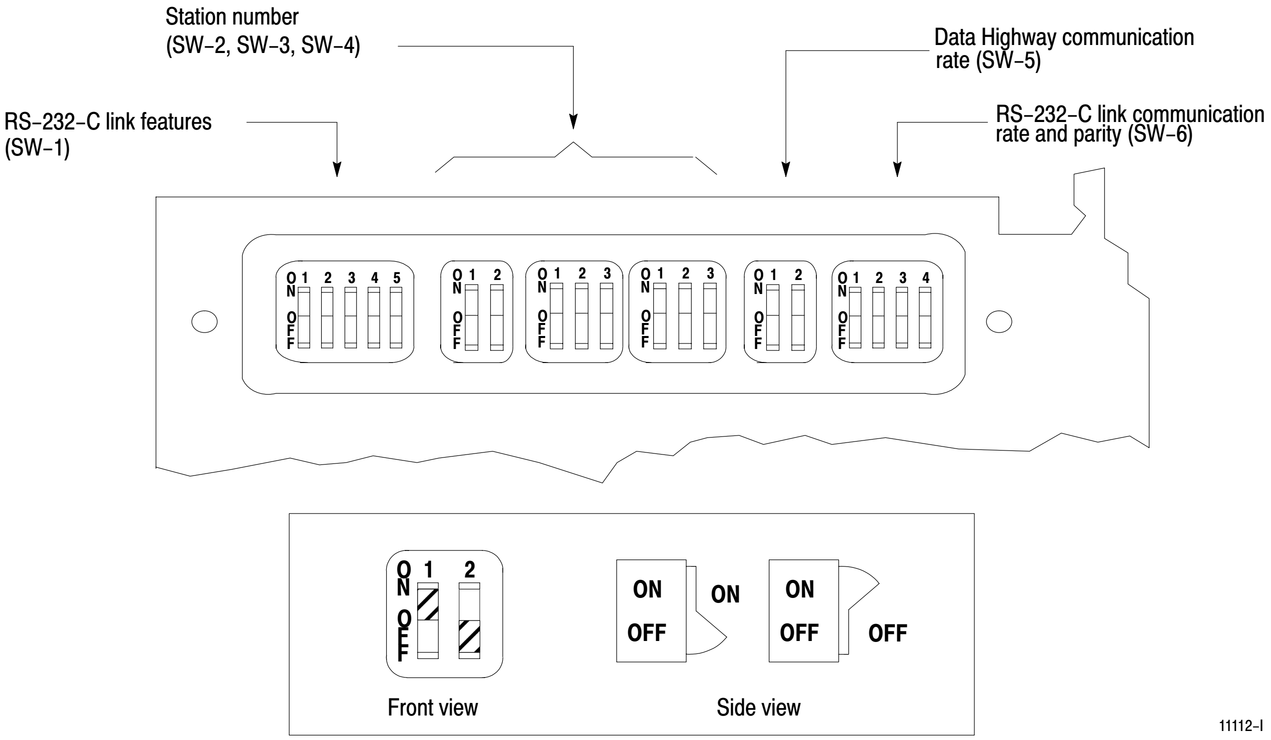
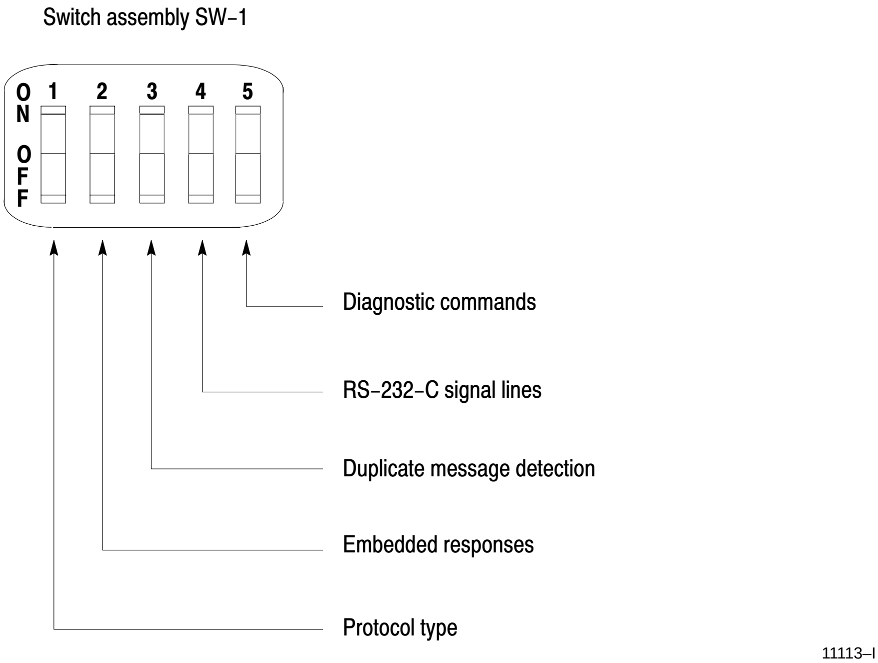


Figure 3.2
RS-232-C Link Features



RS-232-C Link Features Revision A-G

Figure 3.2 illustrates the switches in switch assemblies SW-1. You use these switches to control the special features of the KE/KF module's RS-232-C port.

This section shows you how to control this feature:	using this switch:
protocol	1
embedded responses	2
duplicate message detection	3
RS-232-C handshaking signals	4
diagnostic commands	5

Protocol

Switch number 1 determines what type of protocol the KE/KF module uses in communicating through its RS-232-C port.

If you want your module to have:	set switch 1:
half duplex protocol	on
full duplex protocol	off

This section for revision A-G modules only

Refer to chapter 4 if you need a description of protocols.

Embedded Responses

Switches 1 and 2 determine whether or not the KE/KF module can transmit and receive embedded responses.

If you want your module to:	set switch 1:	set switch 2:
transmit or receive embedded responses	off	on
not transmit or receive embedded responses	on	on
	on	off
	off	off

Refer to chapter 4 if you need a description of embedded responses.

This section for revision A-G modules only

Switch 3 determines whether or not the RS-232-C port of the KE/KF module can detect duplicate messages transmitted to it.

Duplicate Messages

If you want your module to:	set switch 3:
detect and ignore duplicate messages	on
accept all messages regardless of duplication	off

Refer to chapter 4 if you need a description of duplicate messages

Handshaking Signals

Switch 4 determines whether the module uses and recognizes the following handshaking signals: data set ready, request to send, clear to send, data carrier detect, and data terminal ready.

If you want the module's RS-232-C port to:	set switch 4:
use handshaking signals	on
ignore handshaking signals	off

Diagnostic Commands

Switch 5 determines how the KE/KF module treats diagnostic commands sent to it by a remote Data Highway station. You can connect the RS-232-C port of the KE/KF module directly to a 1771-KG, 1773-KA, or 1775-KA communications interface module (Figure 1.2). In such applications, you can set switch 5 so that the KE/KF module will either execute any received diagnostic commands itself or else pass those commands to the other attached communication module.

If you want your module to:	set switch 5:
execute any received diagnostic commands	on
pass any received diagnostic command to the attached RS-232-C device	off

Note that switch 5 relates only to diagnostic commands sent to the KE/KF module from a **remote** Data Highway station. Since only computers can transmit diagnostic commands, the remote station must be a computer connected to the Data Highway by means of another KE/KF module. At the computer station, the setting of switch 5 **does not** affect any diagnostic commands that the computer sends to its local KE/KF module. The local module always retransmits the command message over the Data Highway without executing it. Figure 3.3 illustrates these concepts.

Also note that you can have more than one computer station on a Data Highway, and one computer can transmit diagnostic commands to the others. At the receiving computer station, if switch 5 is off, the local KE/KF module will pass the diagnostic commands to the computer. In such cases, you will have to write computer application programs to handle those commands at the receiving station. If switch 5 is on at the receiving station, the local KE/KF module itself will execute the incoming diagnostic commands.

What to do next

Now skip to section titled “Station Number” to learn how to set station numbers.

RS-232-C Link Features Revision H

The following table shows you how to set the RS-232-C link features for revision H module, switch numbers 1, 2, 5.

If you want to select protocol as:	**with error check as:**	**with parity as:**	**with embedded responses**:	**SW-1**				
				1	**2**	**3**	**4**	**5**
full duplex	BCC	none	no	off	off	N/A	N/A	off
full duplex	BCC	even	no	on	off			off
full duplex	BCC	none	yes	off	on			off
full duplex	BCC	even	yes	on	on			off
half duplex	BCC	none	no	off	off			on
half duplex	BCC	even	no	on	off			on
full duplex	CRC	none	yes	off	on			on
half duplex	CRC	none	no	on	on			on

Figure 3.3
Effect of Switch 5 on Diagnostic Commands

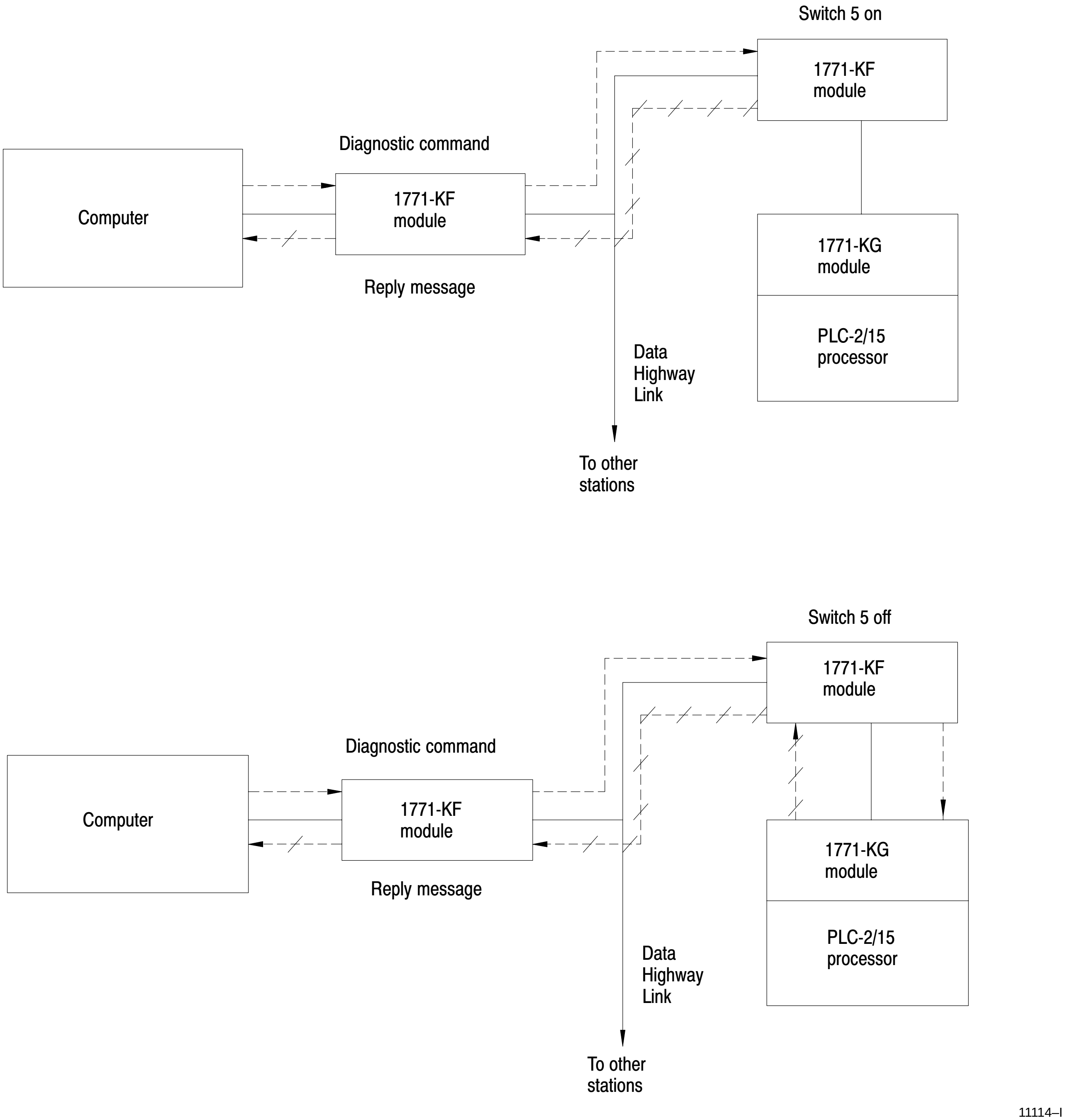
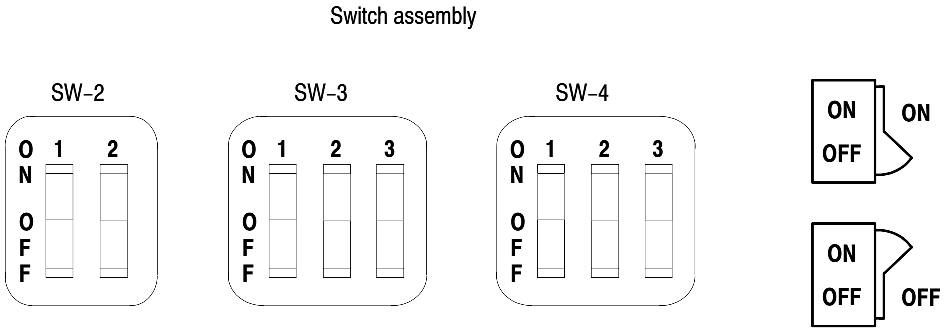


Figure 3.4
Station Number



If you want to set this digit:	Set switches	
	No. 1	No. 2
0	OFF	OFF
1	OFF	ON
2	ON	OFF
3	ON	ON

First Digit

If you want to set this digit:	Set switches		
	No. 1	No. 2	No. 3
0	OFF	OFF	OFF
1	OFF	OFF	ON
2	OFF	ON	OFF
3	OFF	ON	ON
4	ON	OFF	OFF
5	ON	OFF	ON
6	ON	ON	OFF
7	ON	ON	ON

Second and Third Digits

Switch Setting Example: Station No. 037

Switch Assembly	SW–2		SW–3			SW–4		
Station No. Digits	0		3			7		
Switch No.	1	2	1	2	3	1	2	3
Switch Setting	OFF	OFF	OFF	ON	ON	ON	ON	ON

11115–I

This section for the revision H module only

Switch 3 determines whether the module uses and recognizes the following handshaking signals, data set ready, request to send, clear to send, data carrier detect, and data terminal ready.

If you want your module to:	set switch 3:
detect and ignore duplicate messages	on
accept all messages regardless of duplication	off

Switch 4 determines whether or not the RS-232-C port of the KE/KF module can detect duplicate messages transmitted to it.

If you want to the module's RS-232C port to:	set switch 4:
use handshaking signals	on
ignore handshaking signals	off

What to do next

Now go to section titled “Station Number” to learn about station numbers.

Station Number

Switch groups SW-2, SW-3, and SW-4 are for setting the station number of the KE/KF module. The station number is an encoded 3-digit octal number that identifies the KE/KF module as a unique station on the Data Highway. Valid station numbers for the KE/KF module are 010 to 077 and 110 to 376 octal.

Figure 3.4 shows an example of how to set the KE/KF station number to 037 octal. The switches in group SW-2 set the first (left-most) digit of the station number, switch group SW-3 sets the middle digit, and switch group SW-4 sets the last (right-most) digit.

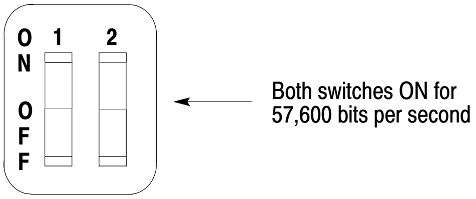
Station numbers play an important part in the polling scheme described in chapter 2. They can also influence the order in which mastership is transferred between Data Highway stations. Therefore, we recommend that you always begin numbering stations with the lowest possible number and continue with the next available number in sequence.

Data Highway Communication Rate

Switch assembly SW-5 lets you select the communication rate for the KE/KF module's Data Highway port.

NOTE: Set both switches ON for a communication rate on the Data Highway of 57,600 bits per second. Be sure to set all Data Highway modules for this communication rate.

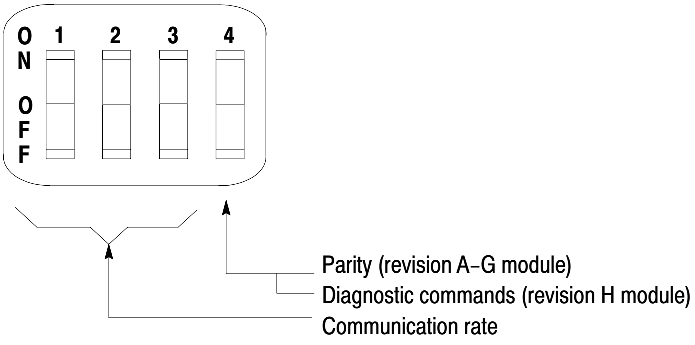
Switch assembly SW–5



RS-232-C Communication Rate, Parity, and Diagnostic Commands

Switch assembly SW-6 lets you select the communication rate, parity, and diagnostic commands for the KE/KF module's RS-232-C port.

Switch assembly SW–6


O
N
O
F
F
1
2
3
4
Parity (revision A–G module)
Diagnostic commands (revision H module)
Communication rate

Communication Rate

You set the communication rate switches the same for both the revisions A-G and H modules.

Bits per	Set switch:		
second as:	1	2	3
110	OFF	OFF	OFF
300	ON	OFF	OFF
600	OFF	ON	OFF
1200	ON	ON	OFF
2400	OFF	OFF	ON
4800	ON	OFF	ON
9600	OFF	ON	ON
19200	ON	ON	ON

NOTE: Any switch settings not shown above will give a communication rate of 9,600 bits per second.

Parity

You can set parity switches only if you hae revision A-G module.

If you want your module to have:	set switch 4:
even parity	on
no parity	off

Diagnostic Commands

To set diagnostic commands for revision A-G modules, refer to the table in the section titled “Diagnostic Commands.”

If you want your module to:	set switcch 4:
execute dignostic commands	on
pass through ddiagnostic ccommands	off

Replacing a 1771-KC/KD Module with a KE/KF Module

By setting the switches properly on the KE/KF module, you can use it to replace an older 1771-KC/KD module in an existing application without having to make any changes in your application programs.

If you have revision A-G module:

In Switch Assembly:	Set Switch Number:	Setting
SW-1	**1**	off
	2	Same setting as 1771-KC/KD switch (assembly SW-1, switch 1)
	3	off
	4	off
	5	Same setting as 1771-KC/KD switch (assembly SW-1, switch 5)
SW-2	**1, 2**	Same setting as 1771-KC/KD switches (assembly SW-2, switches 1 and 2)
SW-3	**1, 2, 3**	Same setting as 1771-KC/KD switches (assembly SW- 3, switches 1, 2, and 3)
SW-4	**1, 2, 3**	Same setting as 1771-KC/KD switches (assembly SW- 4, switches 1, 2, and 3)
SW-5	**1, 2**	Same setting as 1771-KC/KD switches (assembly SW- 5, switches 1 and 2)
SW-6[1]	**1, 2, 3**	Same setting as 1771-KC/KD switches (assembly SW- 6, switches 1, 2, and 3)
	4	off

[1] **NOTE:** The RS-232-C port of the KE/KF module can communicate at a maximum rate of 19,200 bits per second. If your 1771-KC/KD module was set to communicate at a higher rate, then you might have to make some modifications to your RS-232-C link before installing the replacement KE/KF module.

If you have revision H module:

In Switch Assembly:	Set Switch Number:	Setting
SW-1	**1**	off
	2	Same setting as 1771-KC/KD switch (assembly SW-1, switch 1)
	3	off
	4	off
	5	off
SW-2	**1, 2**	Same setting as 1771-KC/KD switches (assembly SW-2, switches 1 and 2)
SW-3	**1, 2, 3**	Same setting as 1771-KC/KD switches (assembly SW- 3, switches 1, 2, and 3)
SW-4	**1, 2, 3**	Same setting as 1771-KC/KD switches (assembly SW- 4, switches 1, 2, and 3)
SW-5	**1, 2**	Same setting as 1771-KC/KD switches (assembly SW- 5, switches 1 and 2)
SW-6[1]	**1, 2, 3**	Same setting as 1771-KC/KD switches (assembly SW- 6, switches 1, 2, and 3)
	4	same settying as 1771-KC/KD switches (assembly SW-1 switch 5)

[1]**NOTE:** The RS-232-C port of the KE/KF module can communicate at a maximum rate of 19,200 bits per second. If your 1771-KC/KD module was set to communicate at a higher rate, then you might have to make some modifications to your RS-232-C link before installing the replacement KE/KF module.

Mounting

The 1771-KE module differs from the 1771-KF in the way it is mounted. The 1771-KE module mounts in an Allen-Bradley Bulletin 1771 I/O rack, while the 1771-KF module is designed for stand-alone mounting.

In both cases, you must mount the KE/KF module within 100 cable feet of the Data Highway truckline. If you are connecting the module directly to an RS-232-C device, you must also mount the module within 50 cable feet of the device. If the RS-232-C device is another Allen-Bradley communication module, you can mount the KE/KF module up to 7,000 away from it by using the longline connection (section titled "Interface Connections"). If you are using a modem link to connect the KE/KF module to the RS-232-C device, then the module and the device may be as far apart as the modem link will allow.

1771-KE Module

To install a 1771-KE module in an Allen-Bradley Bulletin 1771 I/O rack, follow these steps:

1. Turn off all power to the I/O rack and to its controlling PC processor.

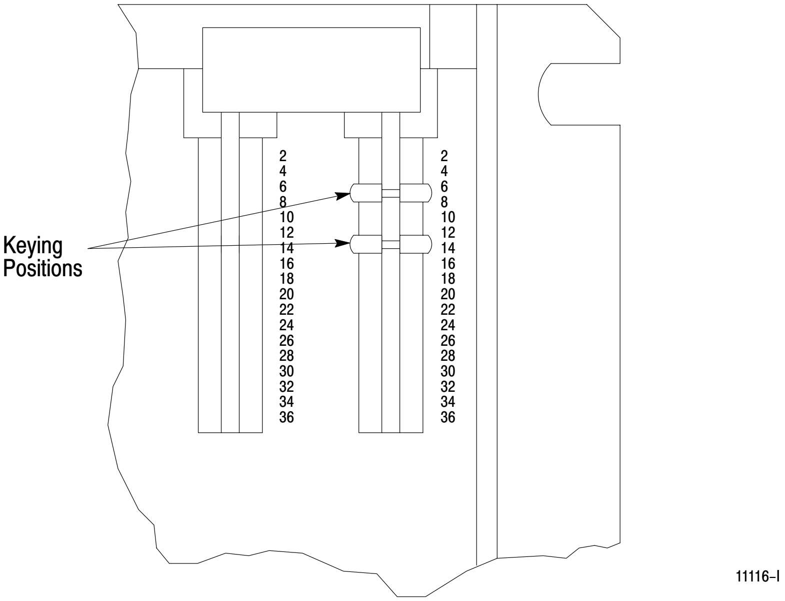
2. Carefully slide the module into one of the slots in the I/O rack.

3. Secure the module in the I/O rack by snapping down the latch on the top of the slot that contains the module.

4. Turn on the power to the PC processor and I/O rack.

Keying

The 1771-KE module is keyed to guard against installation in the wrong module slot. To implement this protection, insert keying bands supplied with your I/O chassis into



You can change the position of keying bands if subsequent system design requires the insertion of a different type of module in this slot.

> **WARNING:** Remove system power before removing or installing your module in the 1771 I/O chassis. Failure to observe this warning could result in damage to module circuitry and/or undesired operation with possible injury to personnel.

1771-KF Module

The rear edge of the 1771-KF module contains a mounting bracket that allows you to mount the module almost anywhere. Figure 3.5 gives the mounting dimensions for the module. To protect the module from harmful environmental effects, enclose it in a standard industrial enclosure (NEMA type 12, or similar).

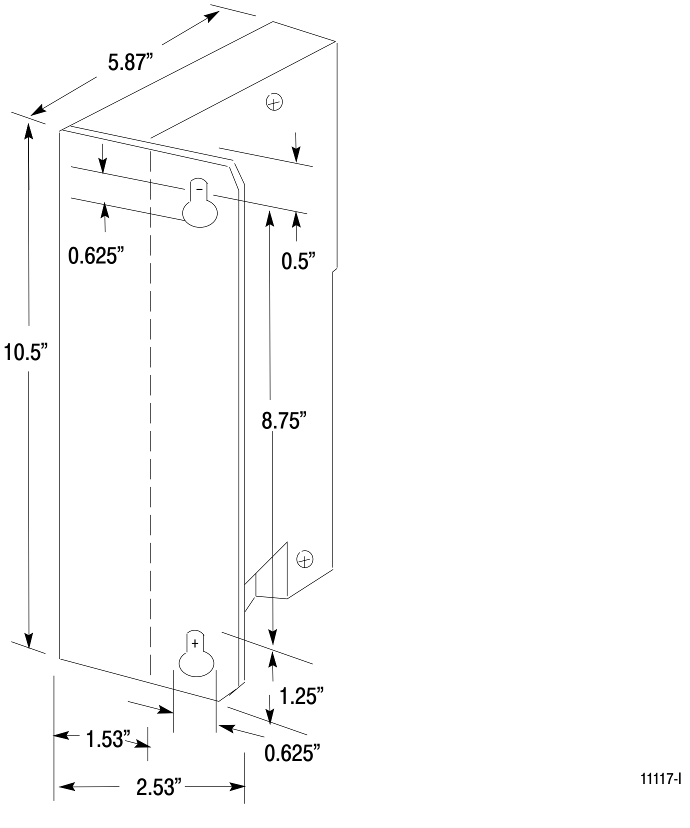
Power Supply

The KE/KF module requires +5V DC power for operation. The 1771-KE module obtains this power from the 1771 I/O rack in which it is mounted. The 1771-KF module needs an independent power supply.

To provide power to a 1771-KF module, use an Allen-Bradley power supply (cat. no. 1771-P2) or equivalent. The power supply connects to a terminal strip at the bottom of the module (Figure 3.6). Use an Allen-Bradley power cable (cat. no. 1770-CF) to make this connection.

Before connecting the 1771-KF module to its power supply, determine whether the supply issues a signal to indicate that its output power is enabled. Some power supplies issue a low-true enable signal, some issue a high-true signal, and others issue no enable signal. The 1771-KF module contains a set of switches that can be set to accept either of these cases. The switches are set incorrectly if all 5 LEDs come on. To set these switches, do the following:

Figure 3.5
Mounting Dimensions for 1771-KF Module



1. Lay the module down so that the white identification label is face down and the front edge of the module is to your right.

2. Remove the screws from the corners of the metal cover place (Figure 3.7).

Figure 3.6
Power Supply Connections for 1771-KF Module

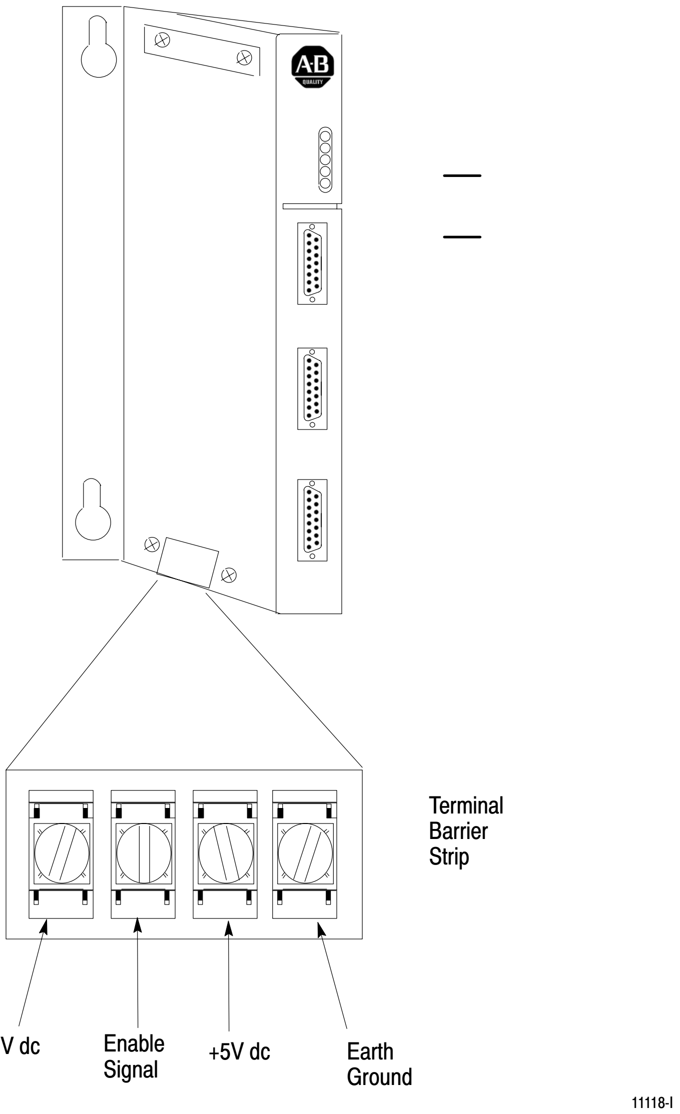
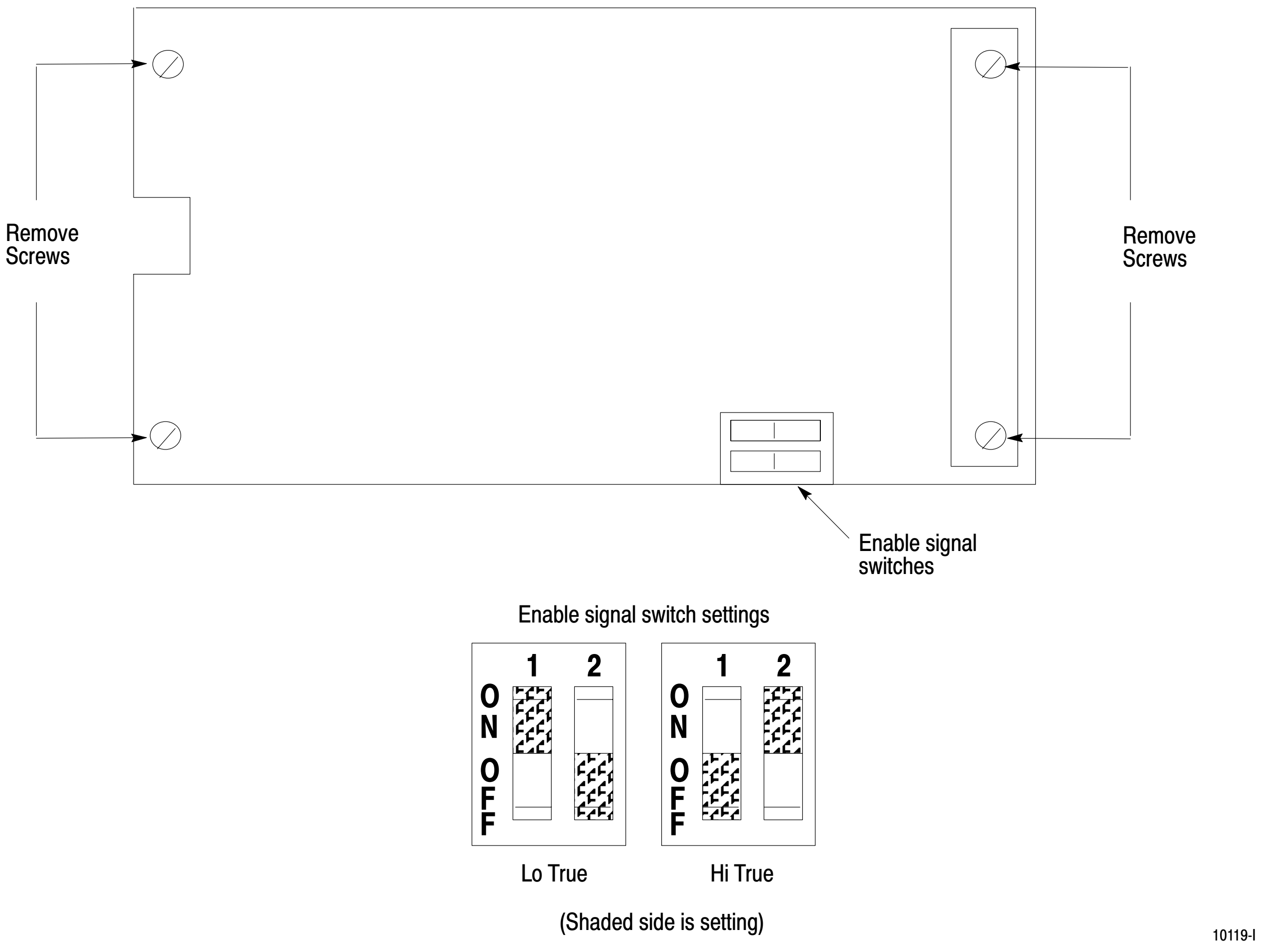


Figure 3.7
Enable Signal Switches for 1771-KF Module



3. Carefully remove the metal cover plate from the module.

4. Locate the enable signal switches on the module circuit board. These switches are situated on the right side of the circuit board, between the indicator lights and the DATA HIGHWAY connector (Figure 3.7).

5. Set the switches for the type of enable signal issued by the power supply. Figure 3.7 shows the settings. Low-true means that the signal line goes low when the supply's output power is enabled; high-true means that the signal line goes high when the supply's output power is enabled. The enable signal must conform to the following specifications:

   High: + 3V to + 5V DC

   Low: -0.2V to -0.6V DC

If the power supply's enable signal does not meet the above specifications, then set both enable signal switches 1 and 2 to the OFF position.

> **CAUTION:** Never set switches 1 and 2 both ON. Doing so disables the 1771-KF module.

If you want this enable signal issued:	Set switch	
	1	2
None	Off	Off
High True	Off	On
Low True	On	Off
Do not use	On	On

6. Replace the metal cover plate and screws.

After setting the enable signal switches, connect the power supply cable to the terminal strip at the bottom of the module (Figure 3.6). Figure 3.6 illustrates the terminal strip, which should be connected as follows:

1. Connect the right-most terminal to earth ground. If the module is mounted inside an enclosure that is already connected to earth ground, then you may connect the right-most terminal to the grounding bus of the enclosure instead.

2. Connect the zero-volt (or ground) lead from the power supply to the first terminal on the left.

3. Connect the +5V DC lead from the power supply to the third terminal from the left.

4. If the power supply has an enable signal line, connect this line to the second terminal from the left.

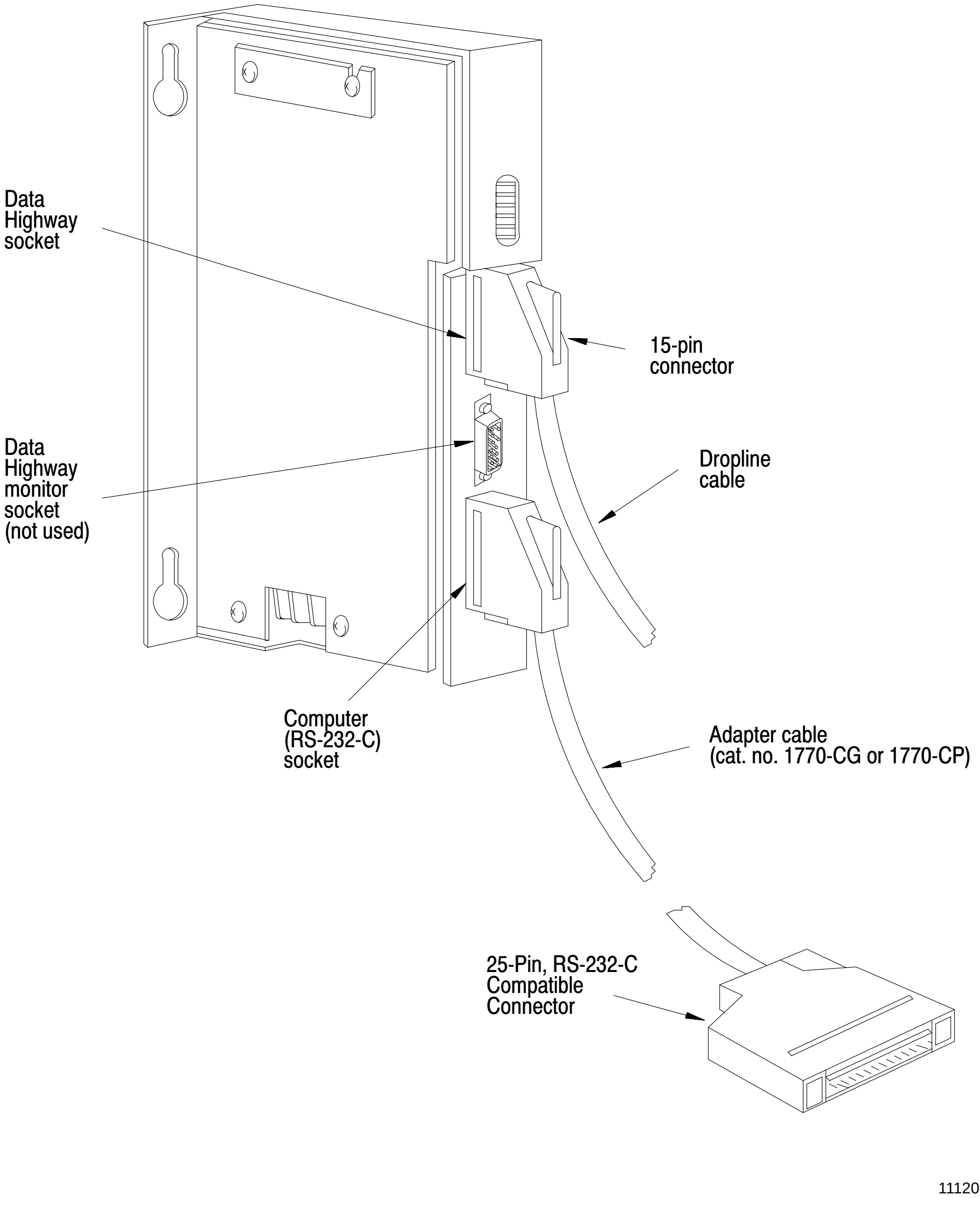
Interface Connections

The KE/KF module has 3 connectors on its front edge (Figure 3.8). The top connector, labeled DATA HIGHWAY, connects to the Data Highway dropline cable. Plug the 15-pin connector of the dropline into the DATA HIGHWAY socket. (For details on how to construct the dropline, refer to publication 1770-810 or 1770-925.)

The center connector, labeled DATA HIGHWAY MONITOR, is for future product development. Do not make any connections to this socket.

The bottom connector, labeled COMPUTER (RS-232-C), connects to an intelligent RS-232-C compatible device. The rest of this section explains how to make connections to this RS-232-C socket.

Figure 3.8
KE/KF Module Connectors


Data Highway socket
15-pin connector
Data Highway monitor socket (not used)
Dropline cable
Computer (RS-232-C) socket
Adapter cable (cat. no. 1770-CG or 1770-CP)
25-Pin, RS-232-C Compatible Connector
11120-I

Mechanical Characteristics

The COMPUTER (RS-232-C) connector on a KE/KF module is a female 15-pin D-shell. Note that this connector does not conform strictly to the RS-232-C standard, which specifies a 25-pin male connector. However, you can use an RS232-C adapter cable (cat. no. 1770-CG or 1770-CP) to go from this connector to a standard 25-pin connector.

Electrical Characteristics

Input and output levels on the RS-232 connector conform to the RS-232-C standard. The transmitter has increased capability to drive an isolated line that is up to 7,000 feet long. The length of this line determines the maximum communication rate on the RS-232-C link, as indicated below.

Line Length in Feet	Maximum Communication Rate in Bits Per Second
Up up 2,000	19,200
2,000 to 4,000	9,600
4,000 to 6,000	4,800
6,000 to 7,000	2,400

The receiver can sense the signal generated by a similar transmitter, and it is electrically isolated from all other circuitry on the module. It consists of an opto-isolater circuit with an input and return line at the RS-232-C connector. All other signals on the RS-232-C connector are driven and received by standard RS-232-C interface circuits, which have maximum drive capability of 50 feet.

Cabling

Cabling for the RS-232-C port of the KE/KF module will vary, depending on your application. In general, the pinouts for this cabling are as follows:

Signal	Abbreviation	Standard RS-232 25-pin Connector	KE/KF Module 15-pin Connector
Chassis/shield drain		1	1
transmitted data	TXD	2	2
received data	RXD	3	3
request to send	RTS	4	4
clear to send	CTS	5	5
data set ready	DSR	6	6
signal ground	GND	7	7
data carrier detect	DCD	8	8
data terminal ready	DTR	20	11
transmitted data return	TXDRET (longlineonly)		14
received data return	RXDRET (longline only)		13

When communication option switch 4 of switch group SW-1 is on (section titled “RS-232-C Link Features Revision A-G”), the RS-232-C port of the KE/KF module can transmit or receive all of the above signals. If communication option switch 4 of switch group SW-1 is off (section titled "RS-232-C Link Features Revision A-G"), then the RS-232-C port uses only TXD, RXD, and GND (and TXDRET and RXDRET for longlines).

The definitions of the above signals are:

- TXD carries serialized data. It is an output from the module.
- RXD is serialized data input to the module. RXD and RXDRET are isolated from the rest of the circuitry on the modules.
- RTS is a request from the module to the modem to prepare to transmit. With full-duplex protocol, RTS is always asserted. With half-duplex protocol, it is turned on when the module has permission to transmit; otherwise it is off.
- CTS is a signal from the modem to the module that indicates the carrier is stable and the modem is ready to transmit. The module will not transmit until CTS is on. If CTS is turned off during transmission, the module will stop transmitting until CTS is restored.
- DTR is a signal from the module to the modem to connect to the phone line (i.e., "pick up the phone"). The module will assert DTR all the time except during the phone hangup sequence. Modems built to American standards will not respond to DTR until the phone rings. Some European modems will always pick up the phone, whether it is ringing or not. The KE/KF module will not work with these types of European modems.
- DSR is a signal from the modem to the module that indicates the phone is off-hook. (It is the modem's answer to DTR.) The module will not transmit or receive unless DSR is on. If the modem does not properly control DSR, or if no modem is used, DSR must be jumpered to a high signal at the module's RS-232 connector. (It can be jumpered to DTR.)
- DCD is a signal from the modem to the module to indicate that the carrier from another modem is being sensed on the phone line. It will not be asserted unless the phone is off-hook. Data will not be received at the RS-232 connector unless DCD is on. With full-duplex protocol, the module will not transmit unless DCD is on. If the modem does not properly control DCD, or if a modem is not being used, DCD must be jumpered to DTR at the module.
- TXDRET is the return signal for TXD. It is connected to module logic ground through a resistor. It does not conform to RS-232-C specifications.
- RXDRET is the return signal for RXD. It is connected to the isolated receiver and is isolated from all other circuitry on the module. It does not conform to RS-232-C specifications.

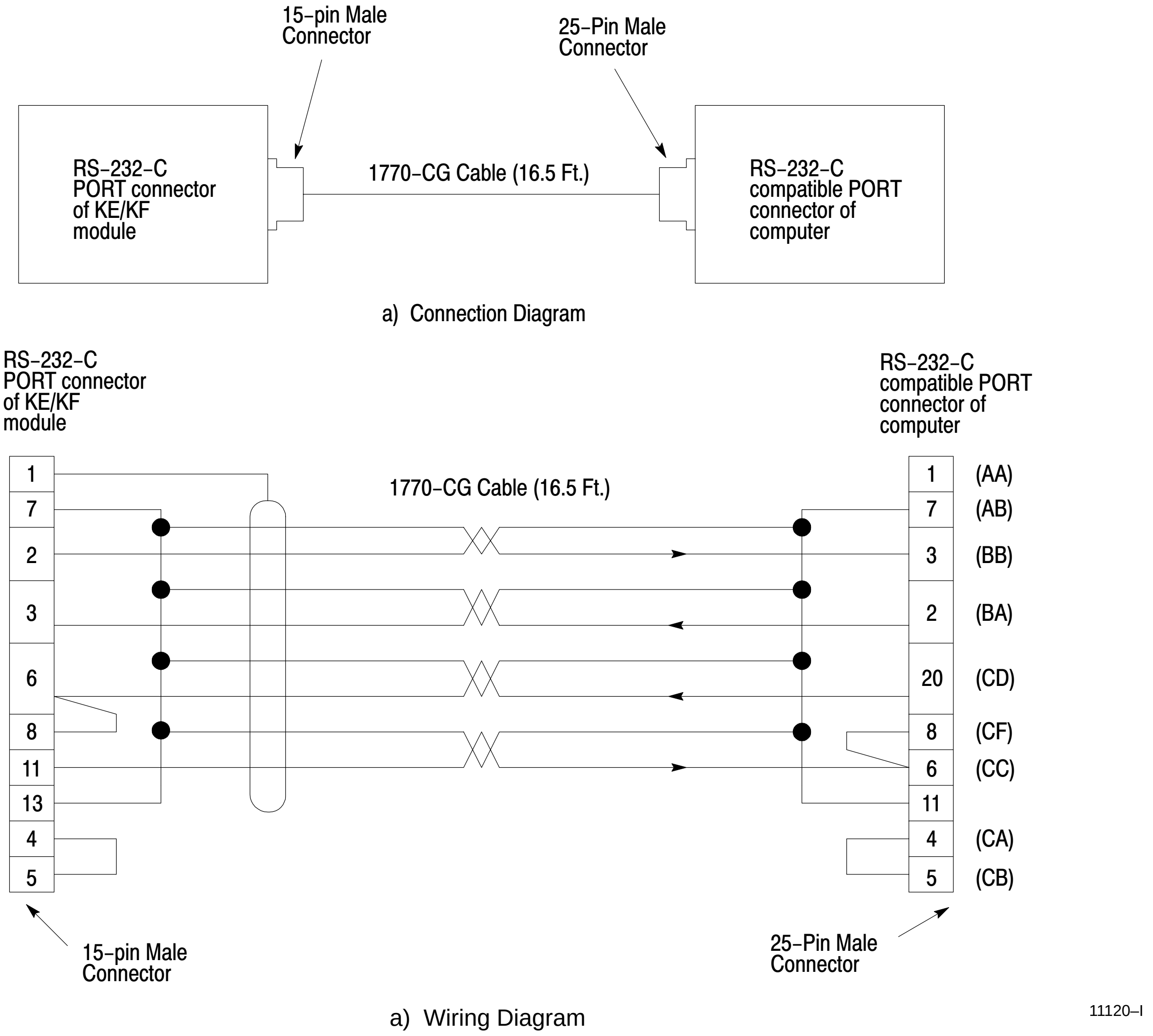
If you are connecting a KE/KF module to a device (e.g., modem or computer) not manufactured by Allen-Bradley, then you must mount the module within 50 cable feet of that device. For such applications, the module's GND must be connected to the GND of the modem or computer. RXDRET must be jumpered to GND at the module. TXDRET should be left open. Note that this type of connection does not provide electrical isolation between the module and the connected device.

Direct Connection to a Computer

To connect the module directly to a computer, you can use a data terminal interface cable (cat. no. 1770-CG). This cable plugs into the COMPUTER (RS-232-C) connector on the module and the RS-232-C compatible connector on the computer (Figure 3.8).

The 1770-CG cable is 16.5 feet long. If you need a long cable or a male/female adapter cable, you can construct your own according to the wiring diagram in Figure 3.9. Connect the cable shield at one end only. Be sure that the cable length does not exceed the RS-232-C limit of 50 feet.

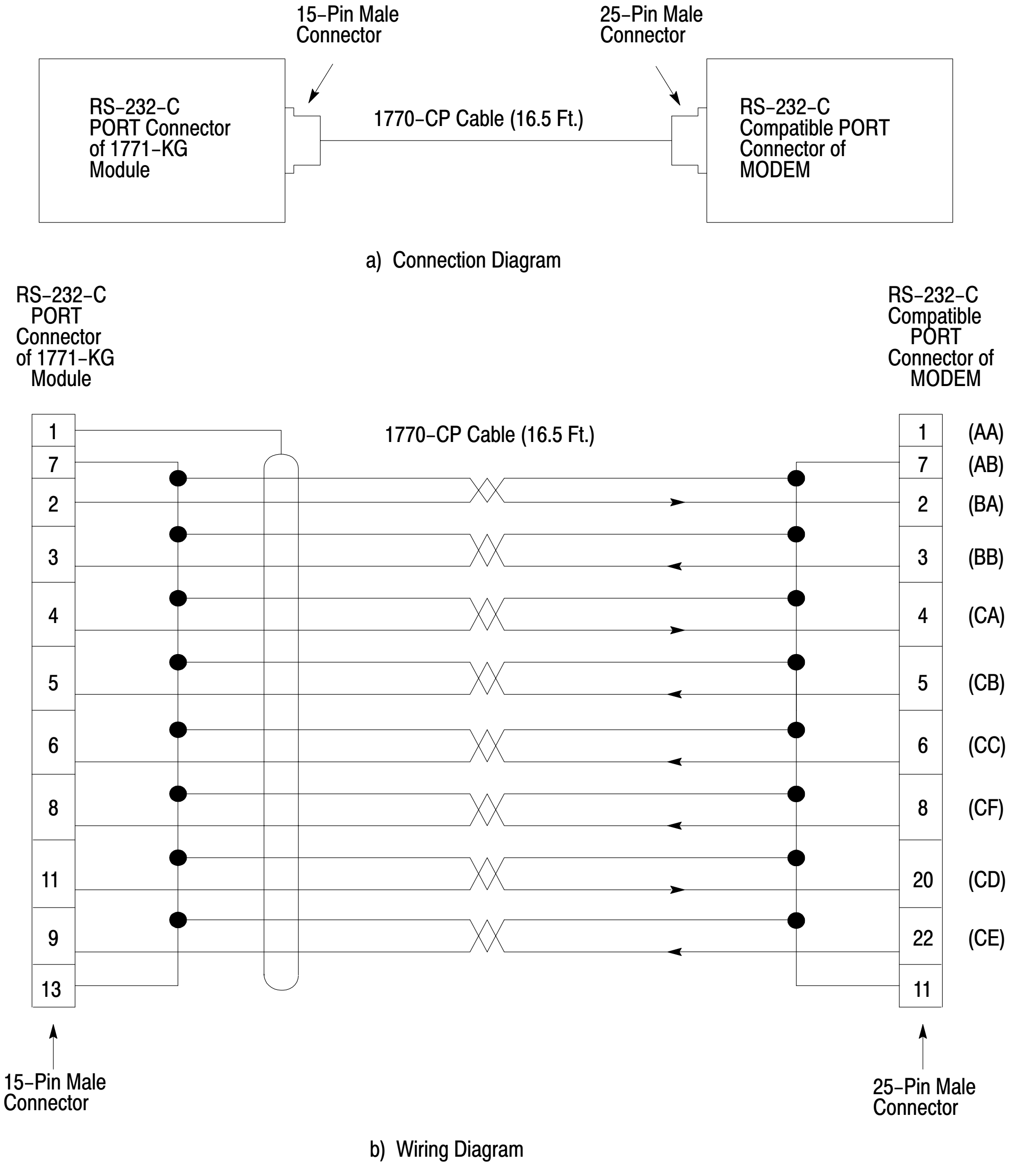
Figure 3.9
Connection to a Computer



a) Connection Diagram

a) Wiring Diagram

Figure 3.10
Connection to a Modem



a) Connection Diagram

b) Wiring Diagram

This type of connection includes the DTR signal to allow each end to detect the loss of the other end's ability to communicate. If your computer does not provide the DTR signal, jumper pins 6 and 8 at the module to pin 11.

Connection to a Modem

To connect the module to a modem, you can use the modem interface cable (cat. no. 1770-CP). This cable plugs into the COMPUTER (RS-232-C) connector on the module and the RS-232-C compatible connector on the modem (Figure 3.10).

The 1770-CP cable is 16.5 feet long. If you need a longer cable or a male/female adapter cable, you can construct your own according to the wiring diagram in Figure 3.10. Connect the cable shield at one end only. Be sure that the cable length does not exceed the RS-232-C limit of 50 feet.

The module can be connected to standard American dial-up modems and some European modems. Other European standards specify that the DTR signal will make the modem answer the phone whether it is ringing or not, causing the phone always to be "busy." Do not use the module with any type of modem that asserts the DTR signal even while waiting for a call.

The types of dial-up network modems that you can use are:

- Manual: these are typically acoustically coupled modems. The connection is established by human operators at both ends, who insert the handsets into couplers to complete the connection.
- DTE-controlled answer: these unattended modems are directly connected to the phone lines. The module serves as the data terminal equipment to control the modem via the DTR, DSR, and DCD signals. The module incorporates timeouts and tests to properly operate these types of modems.
- Auto-answer: these modems have self contained timeouts and tests, and can answer and hang up the phone automatically.

The module has no means of controlling an auto-dial modem, but it can be used in conjunction with a separate auto-dialer.

Connection to Another Communication Module

To provide a longline RS-232-C communication link with a 1771-KG module, refer to Figure 3.11. To provide a longline RS-232-C link with a 1773-KA or 1775-KA module, refer to Figure 3.12.

To construct the cable, use a 15-pin male connector at each end. Use Belden 8723 or equivalent cable (available from Allen-Bradley under cat. no. 1778-CR). Connect the cable shield at one end only.

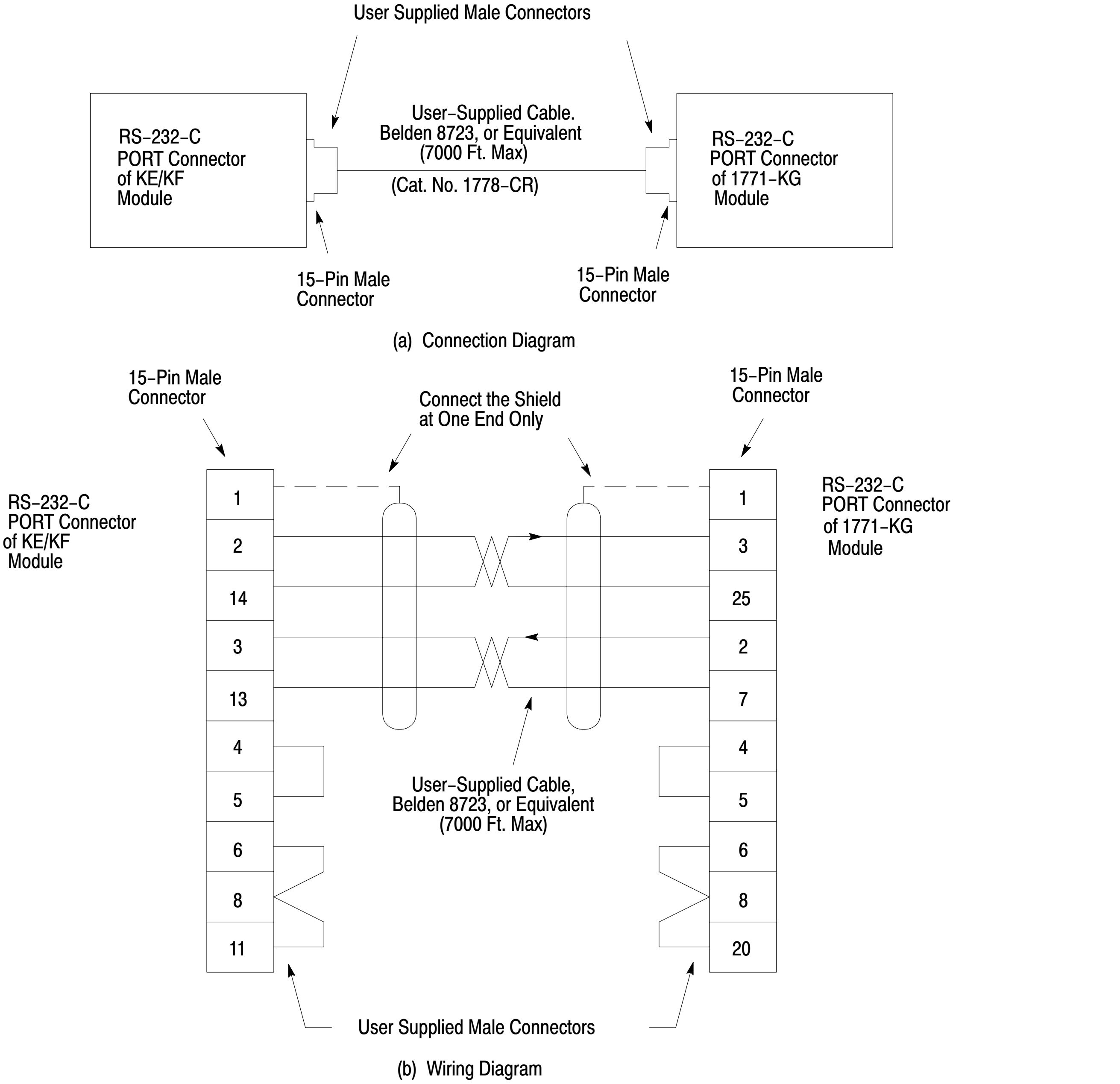
You may make the longline cable up to 7,000 feet long. However, remember that the cable length can limit the communication rate (section titled "Electrical Characteristics").

Answering

The module continually asserts DTR when it is waiting for a call. Under this condition, the modem will answer a call and assert DSR as soon as it detects ringing. The module does not monitor the RING indicator in the RS-232 interface. Once it detects DSR, the module starts a timer (around 10 seconds) and waits for the DCD signal. When the module detects DCD, communication can start.

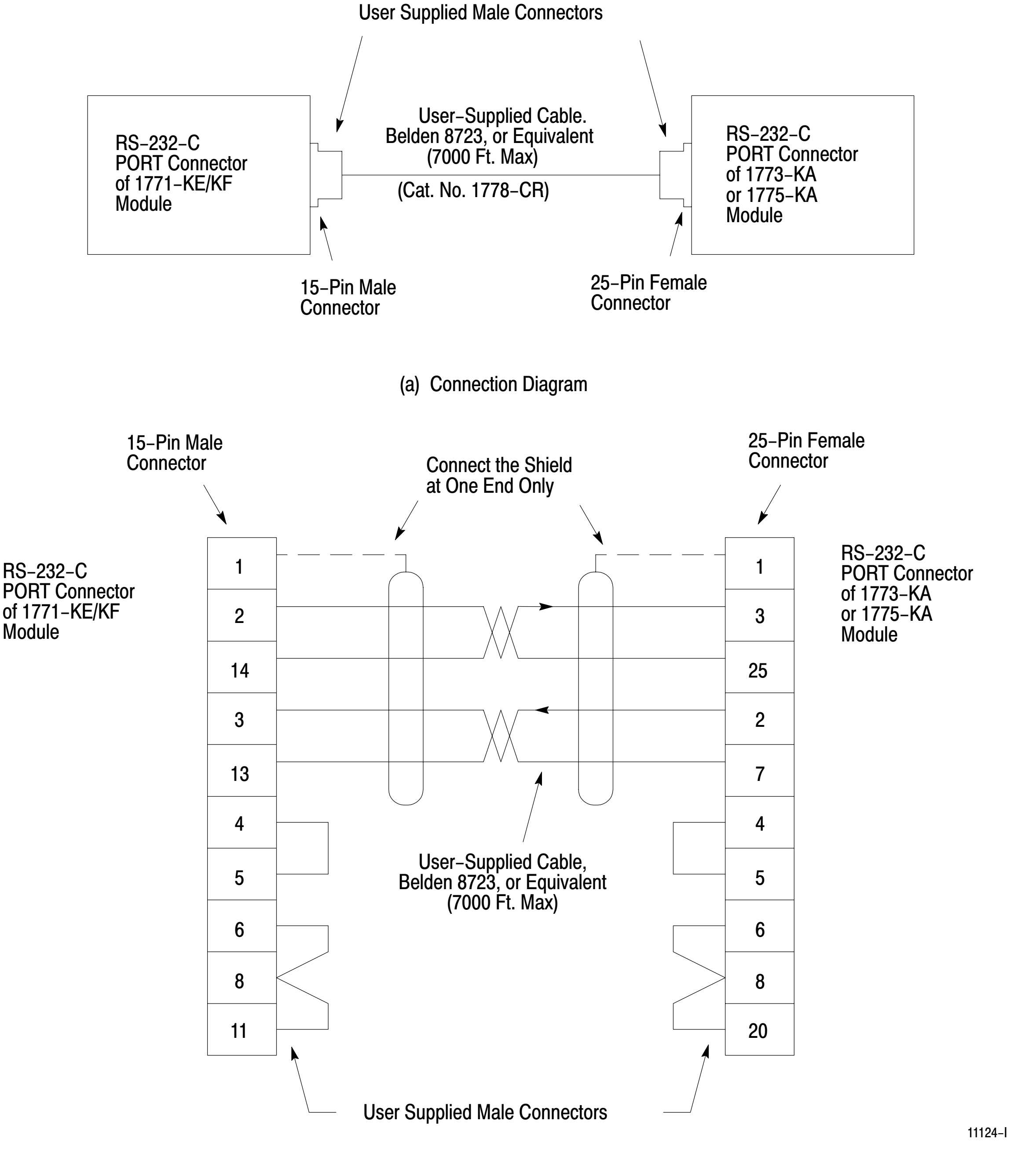
If the module does not detect DCD within the timeout, the module turns DTR off. This causes the modem to hang up and break the connection. When the hang-up is complete, the modem turns off DSR. This causes the module to reassert the DTR line and wait for another call. This feature protects access to the phone if someone calling a wrong number reaches this station.

Figure 3.11
Connection to 1771-KG Module



11123–I

Figure 3.12
Connection to 1773-KA or 1775-KA Module



(b) Wiring Diagram

After detecting DCD, the module continues to monitor the DCD line. If DCD goes off, the module restarts the timeout. If DCD is not restored within the timeout, the module initiates the hangup sequence. This feature allows the remote station to re-dial in the event that the connection is lost through a fault in the phone network.

As soon as DCD goes off, the module responds to any commands that it has received from another data highway station and that it still has stored in its receive buffer. The module's response is to send the same command back to the source station, along with an error code of 84 (chapter 7). At the same time, the module ignores any messages received over its RS-232-C link because it assumes that this link is terminated.

Note that this handshaking is necessary to guarantee access to the phone line. If this handshaking protocol is defeated by improper selection of modem options or by jumpers at the connectors, the modem may still answer a call. But if the connection is lost, the modem will not hang up. It will then be impossible for the remote station to re-establish the connection because it will get a busy signal.

Character Transmission

The module sends data serially over the RS-232-C interface, one 8-bit byte at a time. The transmission format conforms to ANSI X3.16, CCITT V.4, and ISO 1177, with the exception that the parity bit is retained while the data length is extended to eight bits.

The transmission format may be summarized as follows:

start bit
data bit 0
data bit 1
data bit 2
data bit 3
data bit 4
data bit 5
data bit 6
data bit 7
even parity bit (optional)
one stop bit

For communication rate and parity settings, refer to section titled "Data Highway Communication Rate."

Diagnostic Indicators

There are 5 LED indicators on the front of a KE/KF module (Figure 3.13). These indicators can help you in diagnosing problems with the module's installation and operation. The indicators are:

- XMTG - Transmitting
- RCVG - Receiving
- RDY - Ready to transmit

- ACTV - Active
- CU - CPU (NAK indicator)

The XMTG light comes on when the KE/KF module is current master of the Data Highway and is transmitting a command or reply message.

The RCVG light comes on when the module is receiving a command or reply message from another station on the Data Highway.

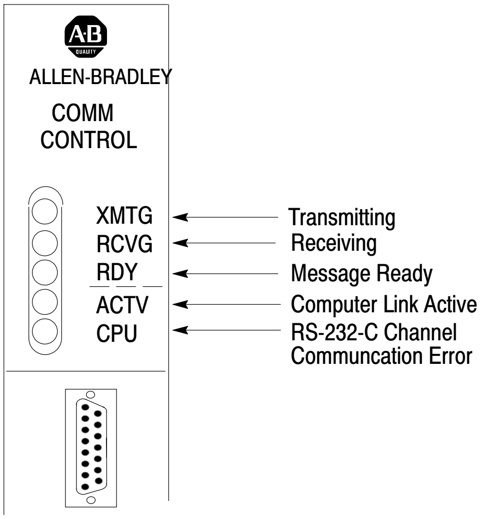
If the XMTG and RCVG lights are on at the same time, this indicates that the module is current master of the Data Highway and is polling the other stations to transfer mastership.

The RDY light comes on when the module has a message stored in its transmit buffer and it is waiting to acquire mastership of the Data Highway so it can transmit the message.

The ACTV light remains on as long as the cable between the COMPUTER (RS-232-C) socket and the interfacing RS-232-C device is properly connected. This light will appear to flicker whenever characters are being transmitted across the RS- 232-C link. If this light goes off, check the cable and connectors for possible problems.

The CPU light comes on for about half a second every time the module transmits or receives a DLE NAK protocol sequence (chapter 4). If this light flickers frequently or stays on, the RS-232-C link might need better isolation or noise immunity.

Figure 3.13
Diagnostic Indicators



Chapter 4

RS-232-C Link Protocols

General

This chapter describes the communication protocol used on the RS-232-C link to the KE/KF module. If you are connecting a KE/KF module to another Allen-Bradley communication interface module (such as a 1771-KG, 1773-KA, or 1775-KA module), then you need not be concerned with protocol because the modules automatically take care of it. However, if you are connecting a KE/KF module to a computer, then you must program the computer to understand and to issue the proper protocol character sequences, as described in chapters 4 through 6 of this manual.

Definition of Link Protocol

A link consists of a wire and associated hardware, such as transceivers, UARTs and error checkers. A link protocol carries a message error-free from one end of the link to the other, or it indicates failure with an error code. Internally it delimits messages, detects and signals errors, retries after errors, and controls message flow. It requires that the link hardware send characters from one end of the wire to the other.

The only purpose of a link protocol is to carry a message intact over a link. It has no concern for the content of the message, the message's function in the operation of higher levels in the system, or the ultimate fate or purpose of the message. Once the message has been reliably carried from one end of the link to the other, the link protocol's concern for that message is ended.

The RS-232-C port of the KE/KF module can use one of two link protocols, which are:

- Full-duplex protocol (for point-to-point communication)
- Half-duplex protocol (for master-slave communication)

In general, full-duplex protocol gives faster data throughput but is harder to implement; half-duplex protocol is easier to implement but gives slower data throughput. Each of these protocols is described independently in sections titled "Full-Duplex Protocol" and "Half-Duplex Protocol," respectively.

NOTE: Some Data Highway documentation might refer to full-duplex and half-duplex protocols as DF1 and polled-mode protocols, respectively.

Full-Duplex Protocol

The full-duplex conforms closely to ANSI X3.28, combining features of subcategories D1 (data transparency) and F1 (two-way simultaneous transmission with embedded responses).

Full-duplex protocol is used over a point-to-point link that allows two-way simultaneous transmission. It is relatively difficult to implement because it requires a system programmer to use interrupts and multi-tasking techniques. It is intended for high performance applications where it is necessary to get the highest possible throughput from the available medium.

Transmission Codes

Full-duplex protocol is a character oriented protocol that uses the following ASCII control characters extended to eight bits by adding a zero for bit 7. See ANSI X3.4, CCITT V.3, or ISO 646 for the standard definition of these characters.

Abbreviation	Hexadecimal Code
STX	02
ETX	03
ENQ	05
ACK	06
DLE	10
NAK	15

The term “code” is defined for use in the following paragraphs as an indivisible sequence of one or more bytes having a specific meaning to the link protocol. “Indivisible” means that the component characters of a code must be sent one after another with no other characters between them. It does not refer to the timing of the characters.

The following codes are used in full-duplex protocol:

Control Codes:

DLE STX
DLE ETX BCC/CRC
DLE ACK
DLE NAK
DLE ENQ

Data Codes:

DATA (single characters having values 00-OF and 11-FF)
DLE DLE (to represent the data 10)

Codes can be grouped into two classes: message codes, which are sent from the transmitter to the receiver, and response codes, which are sent from the receiver to the transmitter.

DLE STX is a message code used to indicate the start of a message.

DLE ETX BCC/CRC is a message code used to terminate a message.

DATA 00-0F and 11-FF are message codes used to encode the corresponding values in the message itself. DLE DLE is a message code used to encode the occurrence of the value 10 (hex) in the message.

DLE ACK, a response code, signals that a message has been successfully received.

DLE NAK, also a response code, signals that an attempt to transfer a message has failed.

DLE ENQ is a message code. It requests the re-transmission of the last receiver code.

Link-Layer Message Packets

A link-layer message packet starts with a DLE STX, ends with a DLE ETX BCC/CRC, and includes all link-layer data codes in between. Data codes can occur only inside a message packet. Response codes can also occur between a DLE STX and a DLE ETX BCC/CRC, but these response codes are not part of the message packet; they are referred to as embedded responses.

Figure 4.1 shows the format of a link-layer message packet for full-duplex protocol, and the layer at which each portion should be implemented. At the end of each message packet is the one-byte BCC field.

Block Check

The block check character (BCC) is a means of checking the accuracy of each message packet transmission. It is the 2's complement of the 8-bit sum (modulo-256 arithmetic sum) of all data bytes between the DLE STX and the DLE ETX BCC/CRC. It does not include any other message packet codes or response codes.

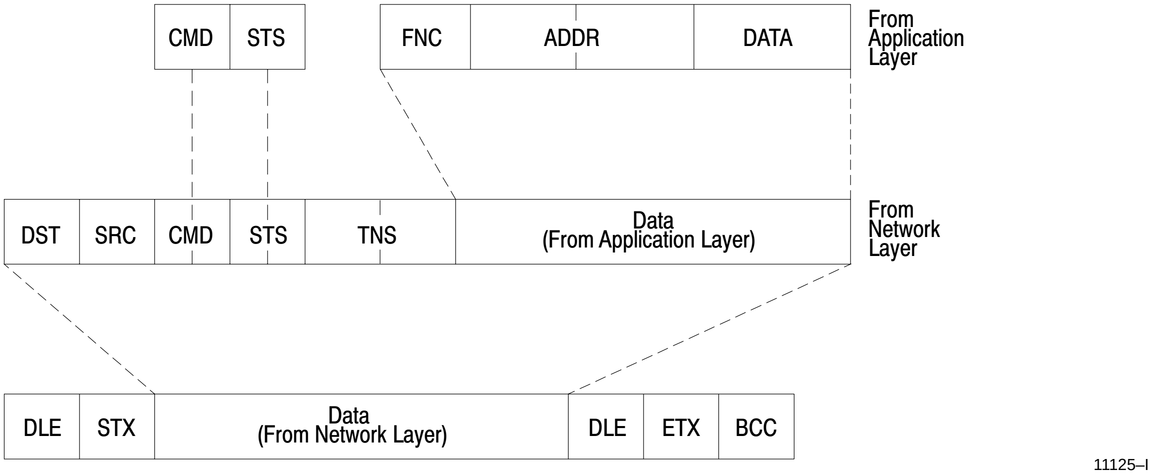
For example, if message packet contained the data codes 8, 9, 6, 0, 2, 4, and 3, the message packet codes would be (in hex):

10 02	08 09 06 00 02 04 03	10 03 EO
DLE STX	Data	DLE ETX BCC/CRC

The sum of the data bytes in this message packet is 20 hex. The BCC is the 2's complement of this sum, or E0 hex. This is shown in the following binary calculation:

```
0010 0000   20 hex
1101 1111   1s compliment
       +1
─────────
1110 0000   2s compliment (E0 hex)
```

Figure 4.1
Link Packet Format for Full-Duplex Protocol



To transmit the data value 10 hex, you must use the data code DLE DLE. However, only one of these DLE data bytes is included in the BCC sum. For example, to transmit the values 8, 9, 6, 0, 10, 4, and 3 hex, you would use the following message codes:

Represents single data byte value of 10

10 02 08 09 06 00 10 10 04 03 10 03 D2

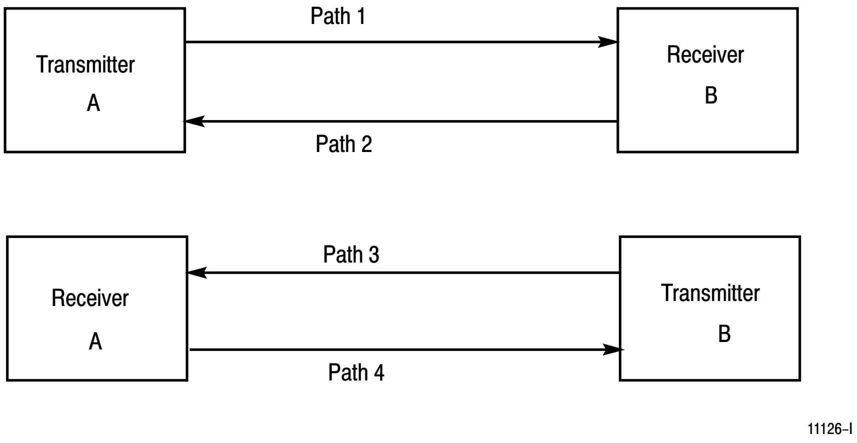
In this case, the sum of the data bytes is 2E hex because only one DLE text code is included in the BCC. So the BCC is D2 hex.

The BCC algorithm provides a medium level of data security. It cannot detect transposition of bytes during transmission of a packet. It also cannot detect the insertion of deletion of data values of zero within a packet.

Two-Way Simultaneous Operation

On a two-way simultaneous link there are only two physical circuits connecting 4 distinct and independent programs. Referring to the diagram below, transmitter A and receiver B manage the transfer of messages from station A to station B by sending message packets from A to B, and returning responses from B to A. At the same time, transmitter B and receiver A carry out the transfer of messages from station B to station A by sending message packets from B to A, and returning responses from A to B. Figure 4.2 shows the four independent data paths involved.

Figure 4.2
Data Paths for Two-Way Simultaneous Operation



Path 1 carries message codes from A to B; path 2 carries response codes from B to A; path 3 carries message codes from B to A; and path 4 carries response codes from A to B.

To implement all these data paths with only two physical circuits, a software multiplexer combines the message codes with the response codes going in the same direction. At the other end of the link a software separator divides the message codes from the response codes. The message codes are sent to the receiver, and the response codes are sent to the transmitter via internal software. On each physical circuit, response codes from a receiver to a transmitter will be seen intermingled with message codes being sent from a transmitter to a receiver. Figure 4.3 depicts this implementation.

Figure 4.4 shows path 1 with unrelated parts of Figure 4.3 removed.

Paths 2, 3, and 4 could be similarly shown.

The full-duplex protocol is symmetrical; that is, anything that can be said about transmitter A, receiver B, and paths 1 and 2 applies equally to transmitter B, receiver A, and paths 3 and 4. There are actually two independent instances of the protocol operating simultaneously. For simplicity the protocol will be defined on the subsystem that carries messages for A to B, with reference to Figure 4.5.

Although the protocols on each subsystem operate independently of each other, there will be a slight interaction as transmission of a message is delayed when a two-character response code is inserted in a stream of message codes. Also, any hardware problem that affects message codes traveling over a hardware circuit will also likely affect response codes on the same circuit.

Figure 4.3
Software Implementation of Data Paths

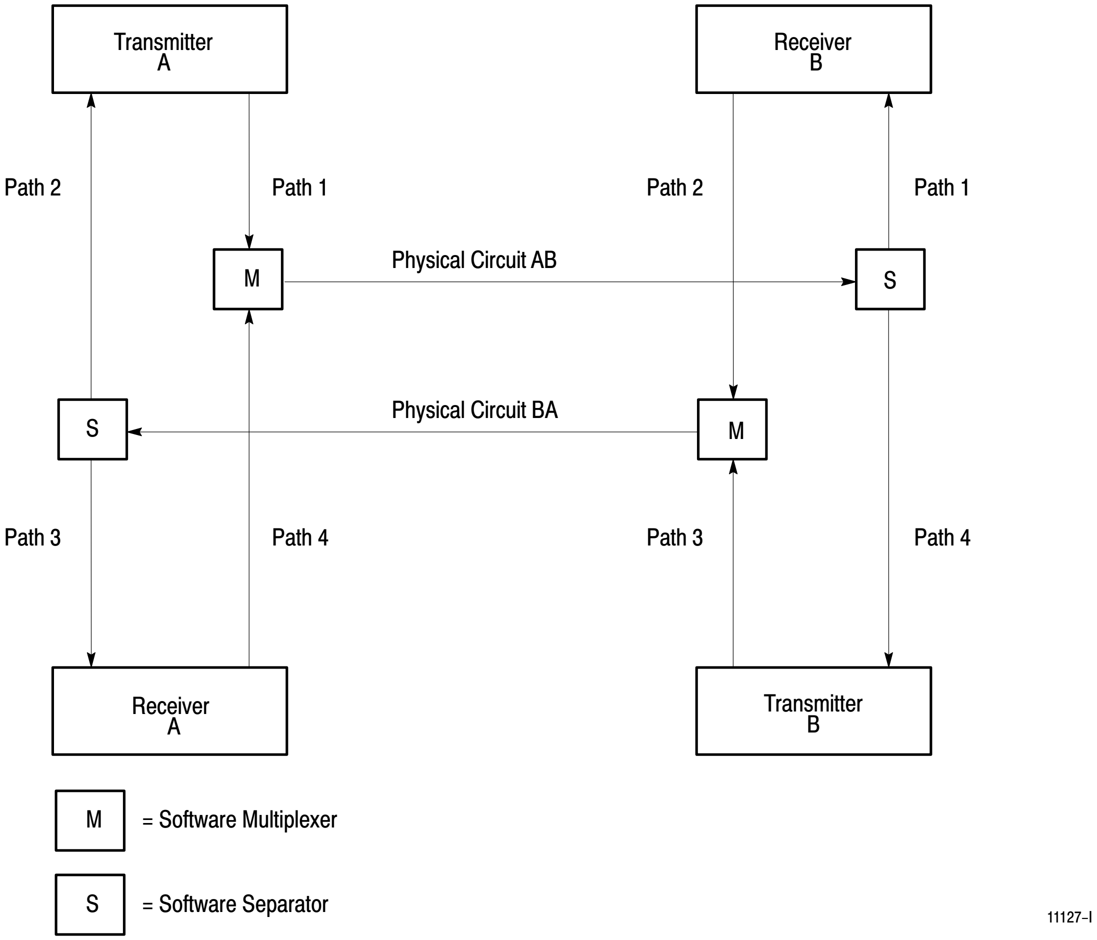


Figure 4.4
Path 1, Unrelated Parts of Figure 4.2 removed

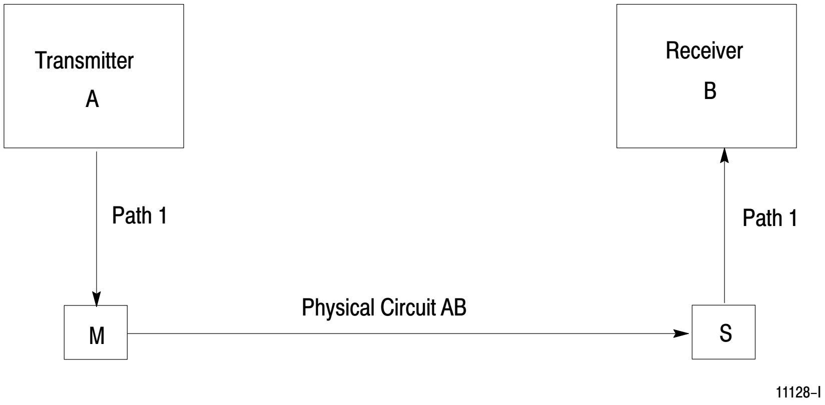
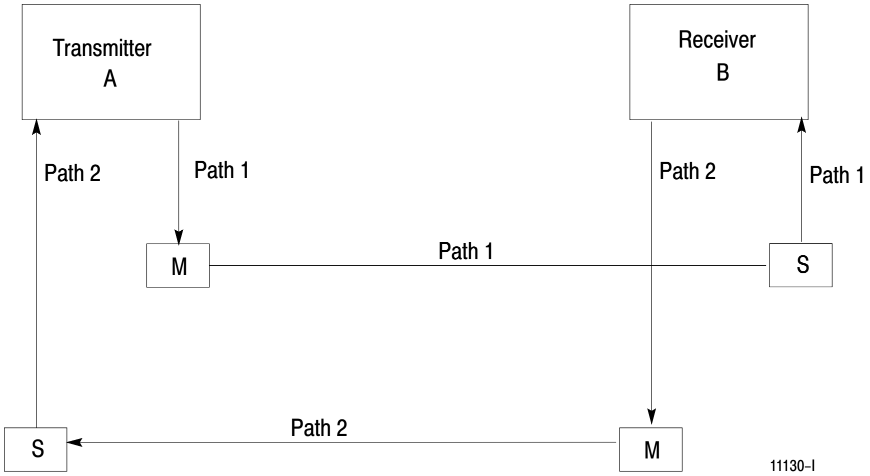


Figure 4.5
Message Transmission from A to B



Protocol Environment Definition

To fully define the environment of the protocol, the transmitter needs to know where to get the message it sends, and the receiver must have a means of disposing of messages. These are implementation dependent functions which shall respectively be called the message source and the message sink.

We assume that the message source supplies one message at a time upon request from the transmitter, and requires notification of the success or failure of the transfer to station B before supplying the next message. When the message source is empty the transmitter waits in an inactive state until a message is available. Whenever the receiver has received a message successfully it attempts to give it to the message sink. The possibility exits that the message sink will be full. The receiver requires an indication of this.

Figure 4.6 shows this representation of the protocol environment.

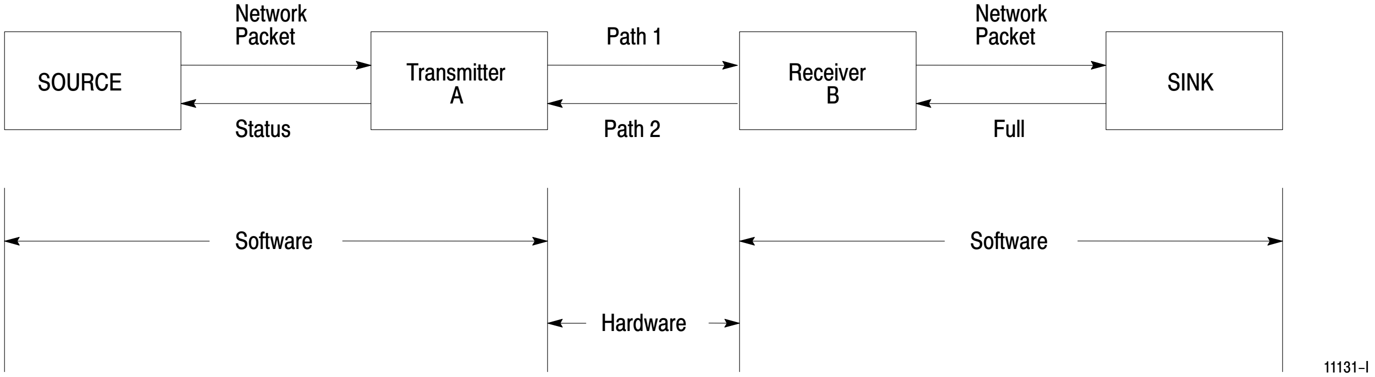
Message Characteristics

Ideally the link protocol should not be at all concerned with the content or form of the message it is transferring. However, full-duplex protocol places the following restrictions on the message that are submitted to it for transfer:

1. Minimum size of a valid message is 6 bytes, maximum is 250 bytes.

2. Some protocol implementations (point-to-point links to a 1771-KG module, for example) require that the first byte of a message match the station address. The receiver will ignore messages that do not contain the correct address.

Figure 4.6
Protocol Environment



3. As part of the duplicate message detection algorithm the receiver checks the second, third, fifth, and sixth bytes of each message. There must be a difference in at least one of these bytes between a message and the previous one for it to be recognized as distinct rather than a retransmission of the previous message. If switch 3 of switch group SW-1 is off, the KE/KF module does not implement duplicate message detection (section titled “RS-232-C Link Features Revisions A-G”).

Protocol Definition

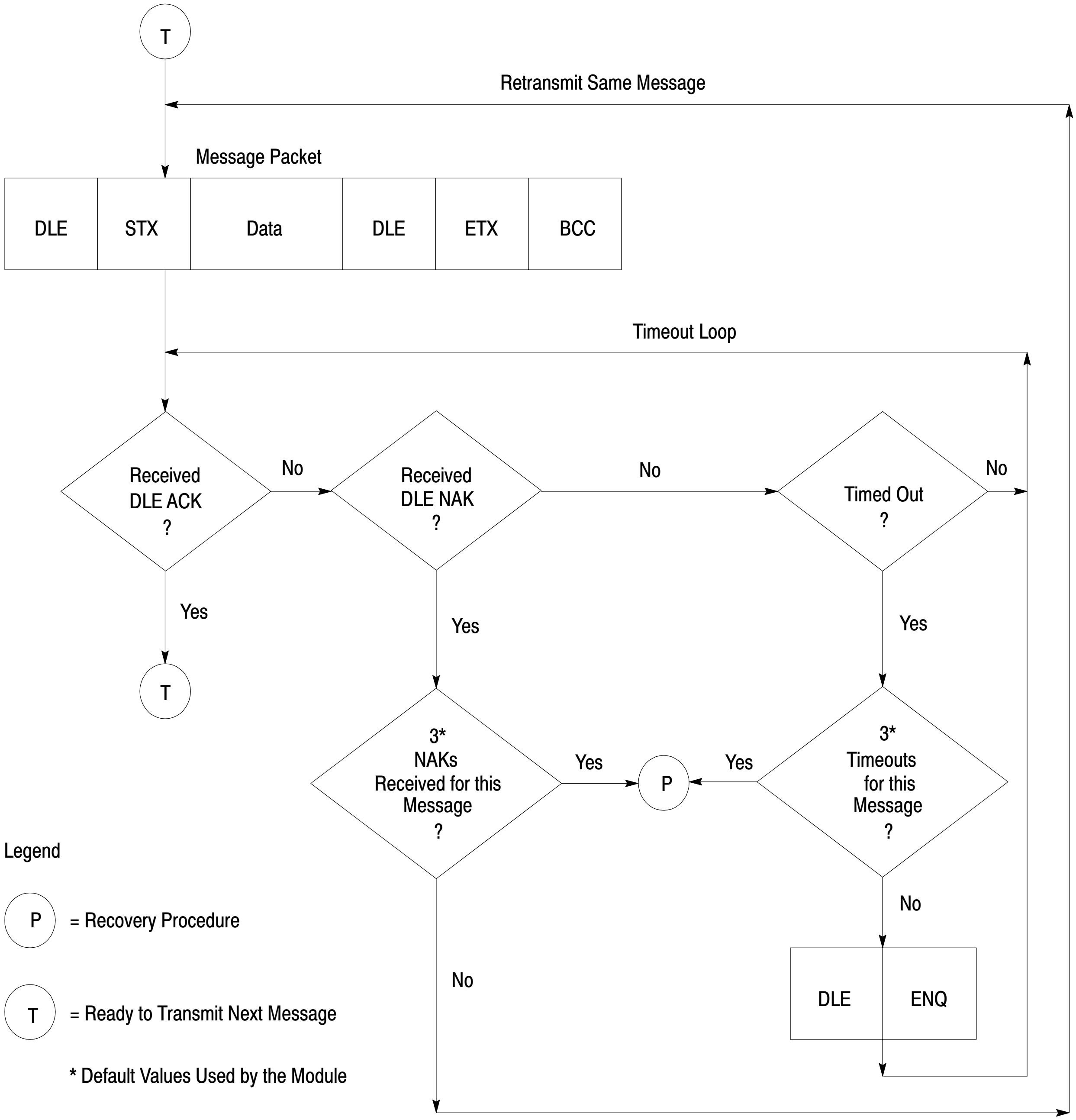
Whenever the message source can supply a message and the transmitter is not busy, it sends a message packet on path 1. It then starts a timeout, and waits for a response on path 2.

When a DLE ACK is received, the message has been successfully transferred. After signaling the message source that the message has been sent, the transmitter proceeds with the next message.

If a DLE NAK is received, the message will be retransmitted. The transmitter restarts the timeout and waits again for a response. This can be repeated several times. There is a user-defined limit to the number of times a message can be retransmitted. If this limit is exceeded, the message source will be signalled of the failure and the transmitter will proceed with the next message.

If the timeout expires before a response is received, the transmitter sends a DLE ENQ on path 1 to request a retransmission of the last response on path 2. It restarts the timeout and waits for a response. This too can be repeated several times, and there is a user-defined limit on the number of timeouts that are allowed per message. If the enquiry limit is exceeded, the message source will be signalled that the transmission has failed, and the transmitter proceeds to the next message.

Figure 4.7
Software Logic for Implementing Transmitter



11122–I

Since there are only two response codes defined, there are no invalid response codes. If the separator returns an invalid response code, the transmitter will ignore it. A more precise and detailed description of the actions of the transmitter appears below in structured English procedures. Figure 4.7 is a flowchart of the software logic for implementing the transmitter.

TRANSMITTER is defined as

```
    loop
    Message=GET-MESSAGE-TO-SEND
    Status= TRANSFER(Message)
    SIGNAL-RESULTS(Status)
    end
```

TRANSFER(Message) is defined as

```
    initialize nak-limit and enq-limit
    SEND(Message)
    start timeout
    loop
    WAIT for response on path 2 or timeout.

    if received DLE ACK then return SUCCESS
    else if received DLE NAK then
      begin
         if nak-limit is exceeded then return FAILURE
        else
             begin
             count NAK re-tries
             SEND-MESSAGE(message);
             end

      end

    else if timeout
       begin
          if enq-limit is exceeded then return FAILURE
         else
             begin
             count ENQ re-tries
             send DLE ENQ on path 1;
             start timeout
             end
       end
    end loop
```

SEND(Message) is defined as

```
    begin
    BCC = 0
    send DLE STX on path 1
    for every byte in the message do
       begin
      add the byte to the BCC;
       send the corresponding data code on path 1
     end

    send DLE ETX BCC on path 1
    end
```

GET-MESSAGE-TO-SEND

This is an operating-system-dependent interface routine that waits and allows the rest of the system to run until the message source has supplied a message to be sent.

SIGNAL-RESULTS

This is an implementation-dependent routine that tells the message source of the results of the attempted message transfer.

WAIT

This is an operating-system-dependent routine that waits for any of several events to occur while allowing other parts of the system to run.

Receiver Actions

Since the receiver gets “dirty” input from the physical world, it is more complex, and must be capable of responding to many adverse situations. Some of the things that can conceivably happen are listed here:

1. The message sink can be full, leaving the receiver with nowhere to put a message.

2. A message can contain a parity error.

3. The BCC can be invalid.

4. The DLE STX or DLE ETX BCC may be missing.

5. The message can be too long or too short.

6. A spurious control or data code can occur outside a message.

7. A spurious control code can occur inside a message.

8. The DLE ACK response can be lost, causing the transmitter to send a duplicate copy of a message that has already passed to the message sink.

A record of the last response sent on path 2 is kept. The value of this response is either ACK or NAK. It is initialized to NAK. When a DLE ENQ is received, the receiver sends the value of the last response from this variable.

A record is kept of several message header bytes. If a message has the same header as the previous message, the message is ACKed but discarded.

The receiver ignores all input from path 1 until a DLE STX or a DLE ENQ is received. If anything other than a DLE STX or DLE ENQ is received on path one, the receiver sets the last response variable to a NAK.

If an ENQ is received, the last response is sent on path 2 and the receiver continues waiting for input.

If a DLE STX is received, the BCC and the message buffer are reset, and the receiver starts building a message.

While building a message, all data codes are stored in the message buffer and added to the BCC. If the buffer overflows, the receiver continues summing the BCC, but the data is discarded. If a parity, overrun, framing, or modem handshaking error is detected, it is recorded. If any control codes other than DLE ETX BCC is received, the message is aborted and a DLE NAK sent on path 2. When the DLE ETX BCC is received, the error flag, the BCC, the message size, and the address (optionally) are all checked. If any of the tests fail, a DLE NAK is sent on path 2.

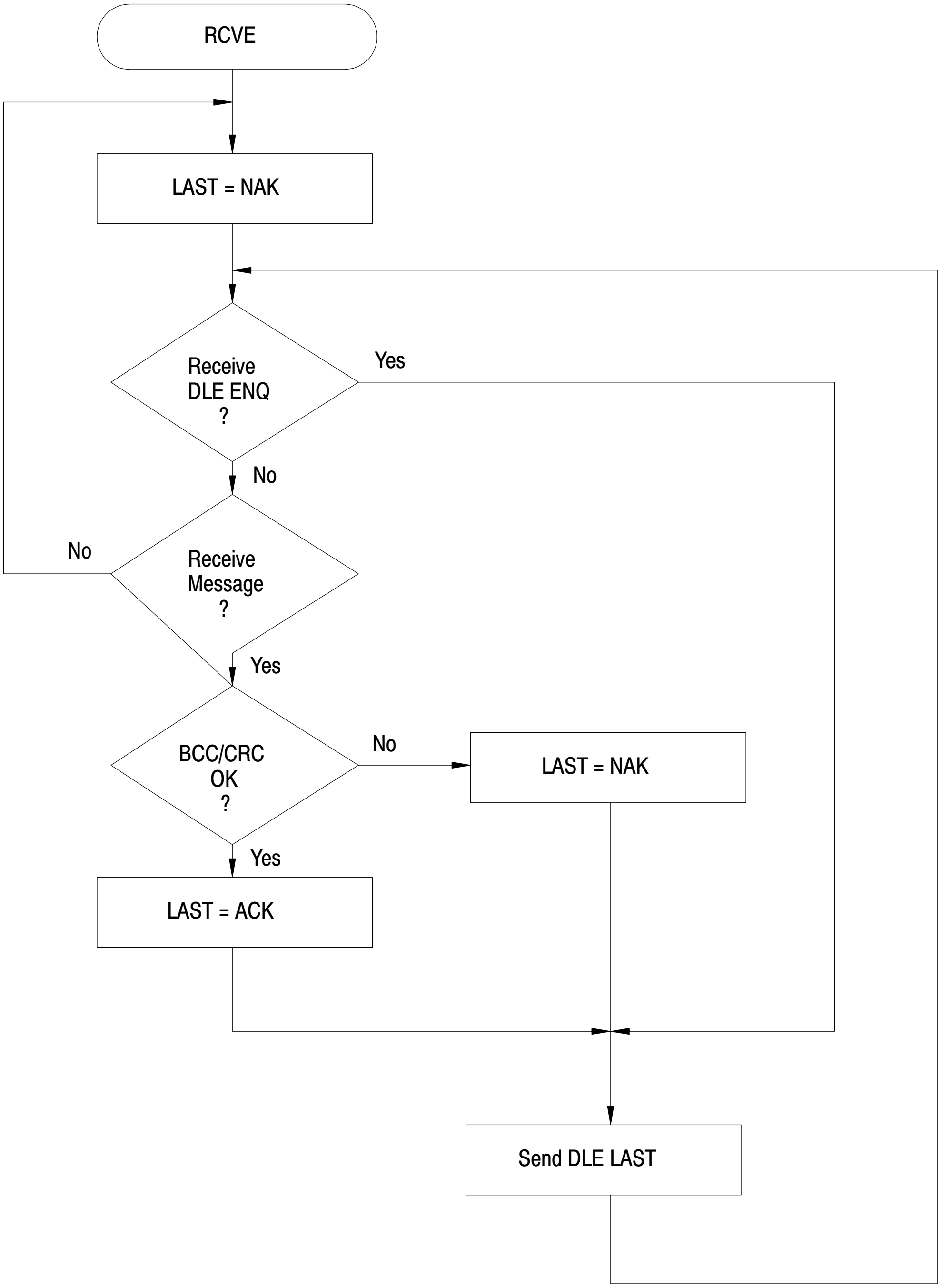
If the message is OK its header is compared to the last message. If it is the same, the message is discarded and a DLE ACK is sent. (Duplicate message detection is not implemented if switch 3 of switch group SW-1 is turned off.)

If the message is different from the last one the state of the message sink is tested. If the message sink is full a DLE NAK is sent; otherwise the message is forwarded to the message sink, the header information is saved for the duplicate message detector, and a DLE ACK is sent.

The procedure for the 17871-KC/KD and 1771-KG(Rev. C) modules is different. First, the check the message sink state. If the sink is full, the response is recorded but not sent. The receiver waits for a DLE ENQ on path 1. If any other code is received, the response is changed from ACK to NAK and the receiver continues waiting for a DLE ENQ. If a DLE ENQ is received, the sink status is checked. If it is still full, the receiver continues waiting. If it is not full, the last response is sent and the receiver then accepts new messages. This variation is not documented in the structured English section below.

The receiver for the KE/KF module is listed below in structured English. Figure 4.8 is a flowchart of the software logic for implementing the receiver.

Figure 4.8
Receiver for Full-Duplex Protocol



11133–I

```
RECEIVER is defined as
  variables
      LAST-HEADER IS 4 bytes copied out of the last good message
      RESPONSE is the value of the last ACK or NAK sent
      BCC is an 8-bit block check accumulator
  LAST-HOLDER  =  invalid
  LAST RESPONSE  =  NAK

  loop
      reset parity error flag
      GET CODE
      if DLE STX then
            begin
                  BCC  =  0
                  GET-CODE
                  while it is a data code
                        begin
                        if buffer is not overflowed put data in buffer
                        GET-CODE
                        and
                  if the control code is not a DLE EXT then send DLE NAK
                  else if error flag is set then send DLE NAK
                  else if BCC is not zero then send DLE NAK
                  else if message is too small then send DLE NAK
                  else if message is too large then send DLE NAK
                  else if header is same as last message send a DLE ACK
                  else if message sink is full send DLE NAK
                  else
                        begin
                            send message to message sink
                            send a DLE ACK
                            save last  header
                        end
            end
      else if DLE ENQ then send LAST-RESPONSE
      else LAST-RESPONSE  =  NAK
  end
GET-CODE is defined as
  loop
      variable
      GET-CHAR
      if char is not a DLE
            begin
                  add char to BCC
                  return the char and data flag
            end
      else
            begin
                  GET-CHAR
                  if char is a DLE
                        begin
                            add char to BCC
                            return a DLE and a data flag
                        end
                  else if char is an ACK or NAK send it to the transmitter
                  else if char is an ETK
                        begin
```

```
                                    GET-CHAR
                                    add char to BCC
                                    return ETX with a control flag
                                 end
                            else return character with a control flag
                       end
                  end
               end
```

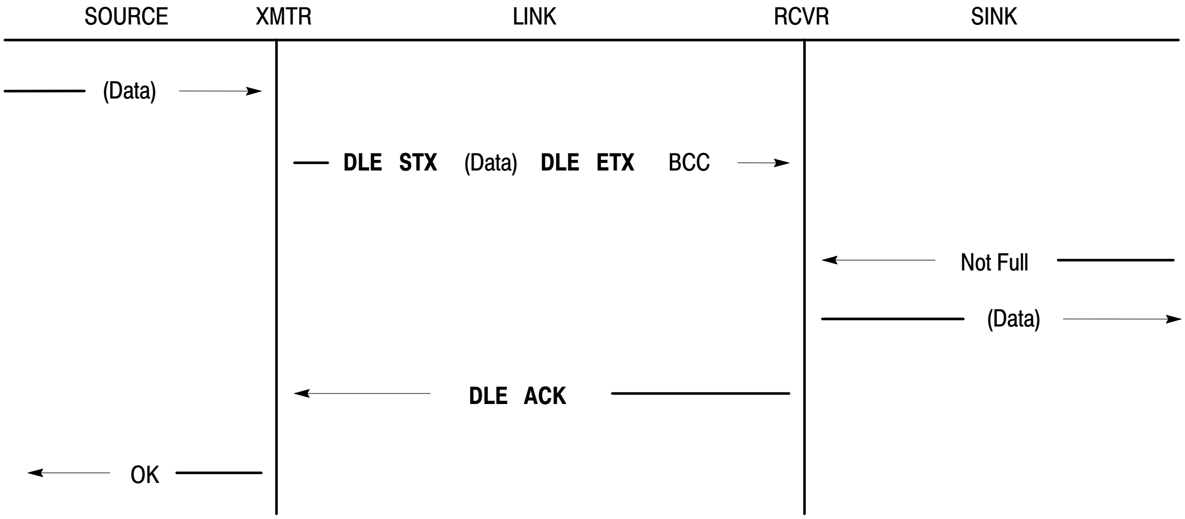
GET-CHAR is defined as

an implementation dependent function that returns one byte of data from the link interface hardware.

Full-Duplex Protocol Diagrams

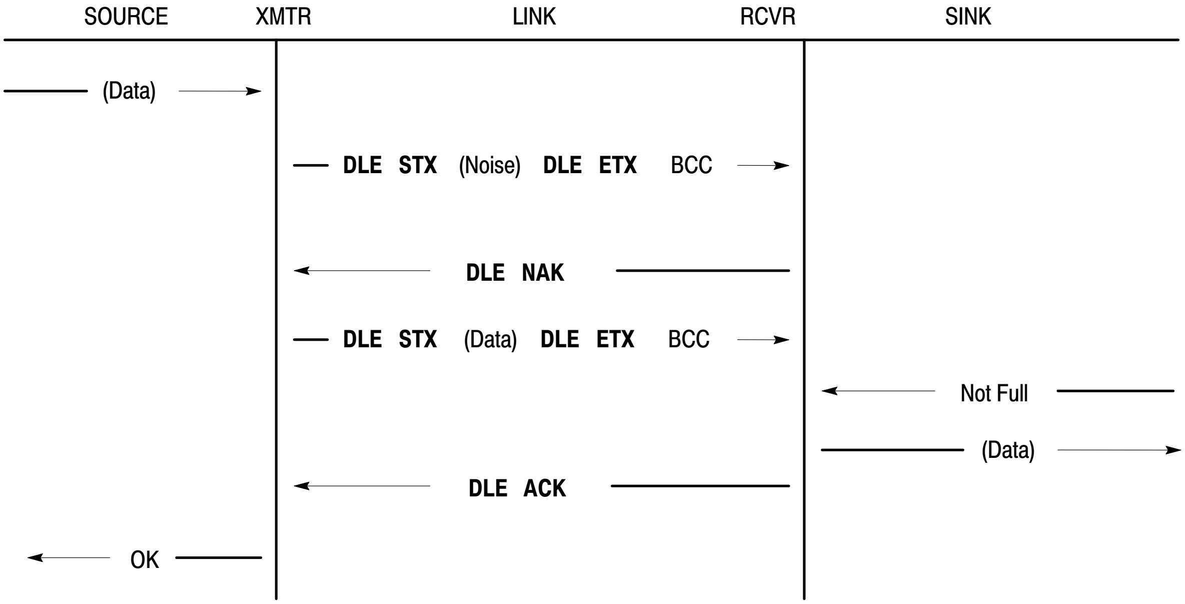
The following figures show some events that can occur on the various interfaces. Time is represented as increasing from the top of the figure to the bottom.

Figure 4.9
Normal Message Transfer



11134–I

Figure 4.10
Message Transfer with NAK



Note that the message sent for the second poll is the same as the first, since the first transmission was not ACKed.

Figure 4.11
Message Transfer with Timeout & ENQ

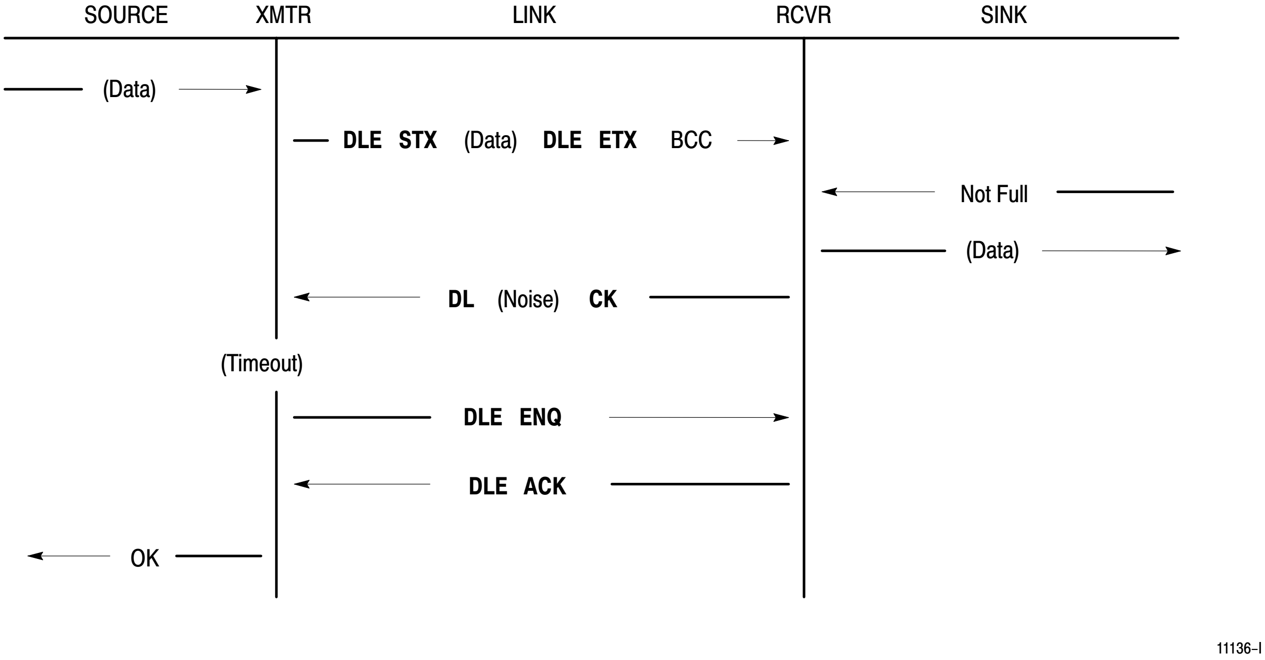
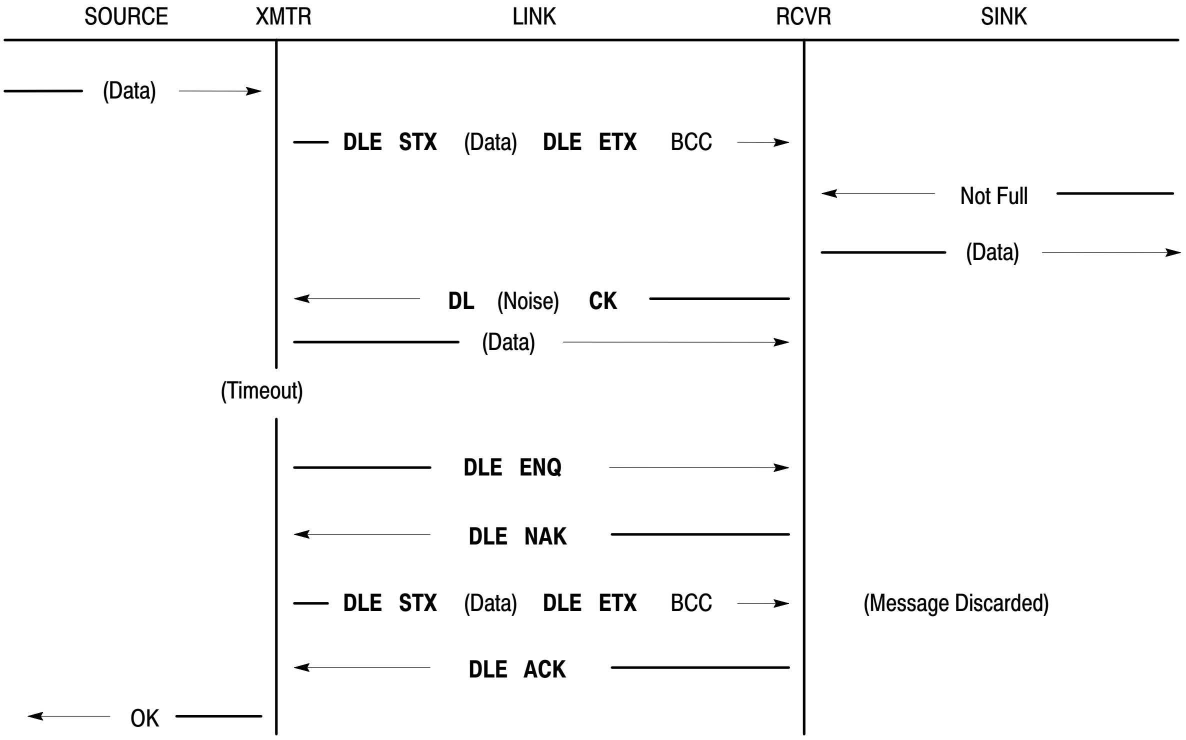


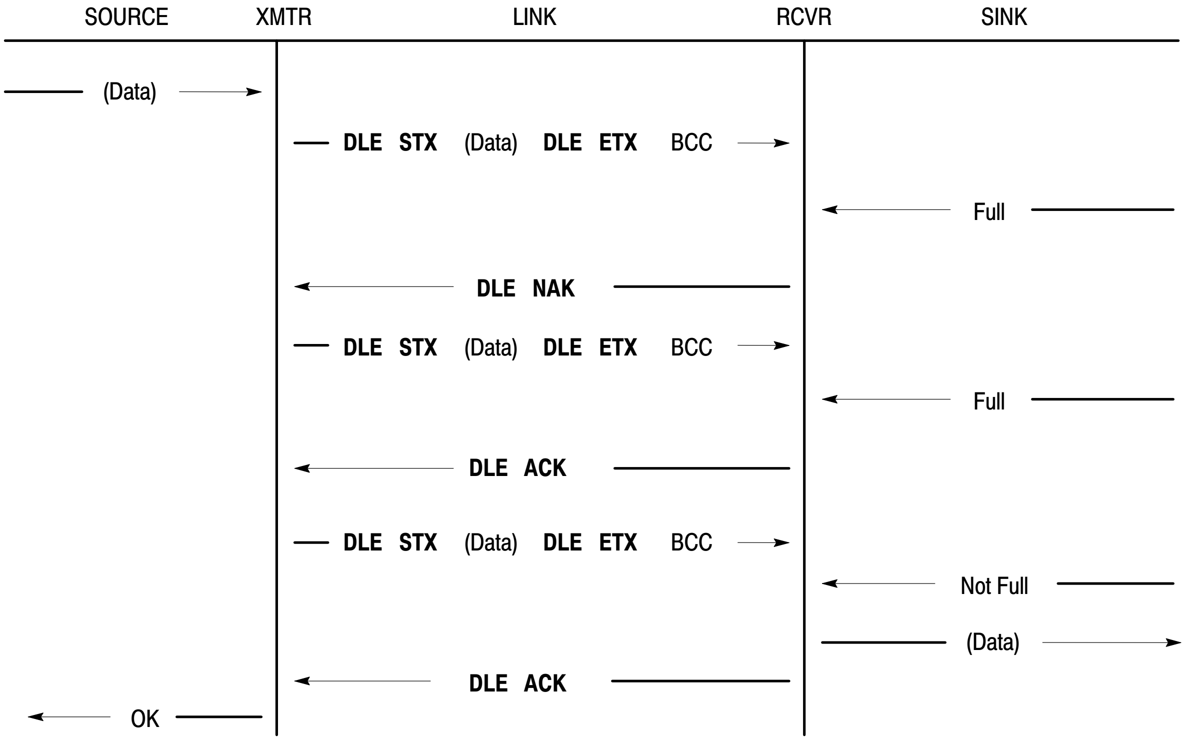
Figure 4.12
Message Transfer with Retransmission



Retransmission occurs when noise hits both sides of the line. This type of noise destroys the DLE ACK while also producing invalid characters at the receiver. The result is that the receiver changes its last response to NAK and the transmitter retransmits the original message. Note that this is detected as a duplicate message if switch 1 of switch group SW-1 is on.

11137-I

Figure 4.13
Message Transfer with Message Sink Full


SOURCE
XMTR
LINK
RCVR
SINK
(Data)
DLE STX (Data) DLE ETX BCC
Full
DLE NAK
DLE STX (Data) DLE ETX BCC
Full
DLE ACK
DLE STX (Data) DLE ETX BCC
Not Full
(Data)
DLE ACK
OK

11138–I

Examples

If a line monitor were connected to the wires between station A and B, and only the A to B subsystem were active, the following would be observed:

Normal message

Path 1: DLE STX...DLE ETX BCCDLE STX....DLE ETX BCC
Path 2: DLE ACK DLE ACK

Message with parity or BCC error and recovery

Path 1: DLE STX..XXX..DLE ETX BCC DLE STX....DLE ETX BCC
Path 2: DLE NAK DLE ACK

Message with ETX destroyed

Path 1: DLE STX....XXXX[timeout] DLE ENQ DLE STX...DLE ETX BCC
Path 2:   DLE NAK   DLE ACK

Good message but ACK destroyed

Path 1: DLE STX...DLE ETX BCC [timeout] DLE ENQ DLE STX...etc.
Path 2:   DLXXXCK   DLE ACK

Messages being sent in both directions

Path 1: DLE STX...DLE ETX BCC   DLE STX....DLE ETX BCC   DLE STX
Path 2:   DLE ACK   DLE ACK

Path 3:   DLE STX...   ....DLE ETX BCC   DLE STX

Path 4:   DLE ACK

Combined -
Circuit AB: DLE STX...DLE ETX BCC   DLE STX....DLE ETX BCC DLE
ACK DLE STX
Circuit BA: DLE STX...DLE ACK....DLE ETX
BCC   DLE ACK DLE STX

embedded response
ACK on AB delayed slightly because ETX BCC are indivisible

Embedded Response Option

To allow simplification of the design of the receiver in some cases, you can disable transmission of embedded responses by turning off communication option switch 2 of switch group SW-1. If this switch is off, the KE/KF module's multiplexer will not allow response codes to be sent within a message. Instead, it will delay response codes until after it receives the next DLE ETX BCC sequence.

Half-Duplex Protocol

Half-duplex protocol is an alternate link protocol to full-duplex protocol. You can select this protocol by turning on communication option switch 1 of switch assembly SW-1. Half-duplex protocol is based on full-duplex but extends or redefines several features.

Half-duplex protocol is a multidrop protocol for one master and one or more slaves. Modems must be used (unless there is only one slave). The units have slave mode capability only; the master function must currently be provided by a user-programmed intelligent device.

Half-duplex protocol provides a lower effective utilization of resources than full-duplex, but it is much easier to implement. Its use is indicated by the following:

- Multiple stations and a single computer are connected in a multidrop configuration using multi-drop modems.
- Half-duplex modems are being used.
- It is desirable to sacrifice throughput in exchange for ease of implementation.

Half-duplex protocol requires the following communication characteristics:

- 8 bits per character
- Even parity
- 1 stop bit
- Pass-all mode (the terminal driver does not translate or respond to control characters)
- Single character buffer (the terminal driver immediately returns each character to the caller)

Multidrop Topology

The intended environment for half-duplex protocol is a multidrop link with all stations interfaced through half-duplex modems. The actual nature of the link does not matter much, as long as the modems support request-to-send, clear-to-send, and data-carrier-detect. If dial-up modems are used they must also support data-set-ready and data-terminal-ready, otherwise DSR should be jumpered to DTR.

There may be from 2 to 256 stations simultaneously connected to a single link. Each station has a receiver permanently connected to the circuit, and a transmitter that may be enabled or disabled by RTS.

One station is designated as master and controls which station has access to the link. Since Allen-Bradley currently does not supply the master station, this function must be performed by your own programmed intelligent device. All other stations are called slaves, and must wait for permission from the master before transmitting. Each slave station has a unique station number from 0 to 254.

The master can send and receive messages to and from each station on the multidrop. The master can send and receive messages to and from every station on highways that are connected to the multidrop via a KE/KF module. If the master is programmed to relay messages, then stations on the multidrop can engage in peer-to-peer communications. The multidrop will not easily provide flexible peer-to-peer communication between the multidrop and connected highways, or between different highways.

Either a two-circuit system (master sends and slaves receive on one circuit, slaves send and master receives on the other), or a one-circuit system (master and slaves send and receive on the same circuit) may be used.

Half-duplex dialup modems can be used as long as a carrier is detected before the carrier timeout (about 8 seconds). If a carrier is not sensed before the timeout the module will hang up the phone. Carrier must be sensed at least every 8 seconds to maintain the connection.

Multiple masters are not allowed, except when one acts as a backup to the other, and does not communicate unless the primary is shut down.

Transmission Codes

Half-duplex protocol is a character oriented protocol that uses the following ASCII control characters extended to 8 bits by adding a zero for bit 7. See ANSI X3.4, CCITT V.3, or ISO 646 for the standard definition of these characters.

Abbreviation	Hexadecimal Code
SOH	01
STX	02
ETX	03
EOT	04
ENQ	05
ACK	06
DLE	10
NAK	15

The term “code” means (in the following paragraphs) an indivisible sequence of one or more bytes having a specific meaning to the link protocol. “Indivisible” means that the component characters of a code must be sent one after another with no other characters inserted between them. It does not refer to the timing of the characters. (This definition has less significance than for full-duplex protocol, since there is no multiplexing of transmission codes in half-duplex protocol).

These codes are used in half-duplex protocol:

Control Codes:

DLE SOH
DLE STX
DLE EXT BCC/CRC
DLE ACK
DLE NAK
DLE ENQ
DLE EOT

Data Codes:

DATA (single characters having values 00-0F and 11-FF)
DLE DLE (to represent the data 10)

DLE SOH indicates the start of a message.

DLE STX separates the link level header from the data field of a message.

DLE ETX BCC/CRC terminates a message.

DATA 00-OF AND 11-FF encode the corresponding values in the message itself. DLE DLE encodes the occurrence of the value 10 (hex) in the message.

DLE ACK signals that a message has been successfully received.

DLE NAK is used as a global link reset command. This causes all slaves to cancel all messages that are ready to be transmitted to the master. Typically the slave will return the messages to the command originator with an error code.

DLE ENQ starts a poll command.

DLE EOT is used by slaves as a response to a poll when they have no messages to send.

Link-Layer Packets

Half-duplex protocol uses three types of transmissions:

- Polling packet
- Master message packet
- Slave message packet

The master station transmits both polling packets and master message packets, while slave stations transmit slave message packets.

Figure 4.14 illustrates the formats of these packets. Note that the slave message packet has the same format as the full-duplex message packet (section titled

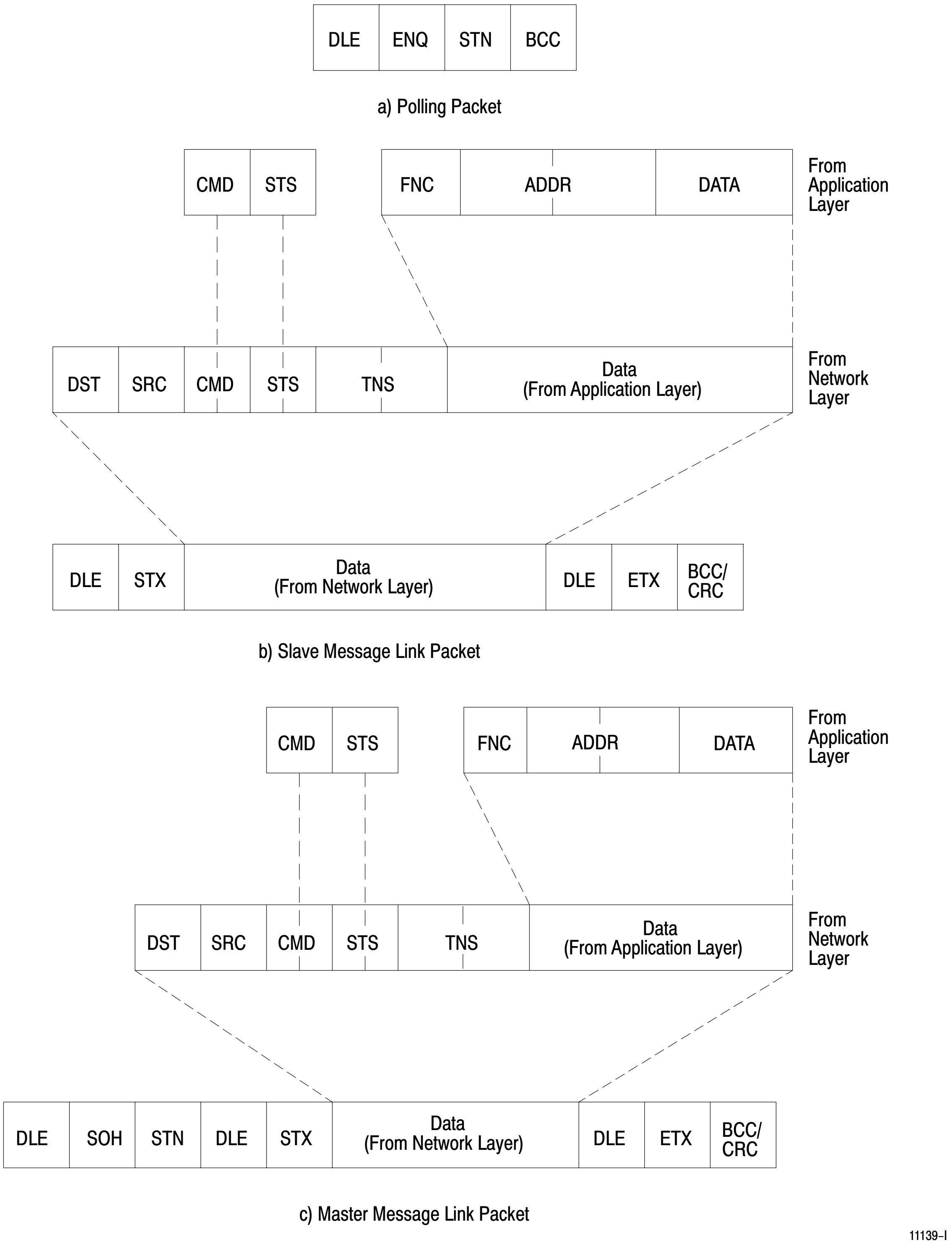
“Link-Layer Message Packets”). The master message packet is the same as the slave message packet except that it is prefixed with DLE SOH and an address code to specify a slave station number.

At the end of each polling packet, there is a BCC byte. At the end of each message packet, there is either a one-byte BCC field, or a two byte CRC field. With a series A-G module, you must use BCC. With series H module, you can select BCC or CRC through switch settings.

Block Check

The block check character (BCC) is a means of checking the accuracy of each packet transmission. It is the 2’s complement of the 8-bit sum (modulo-256 arithmetic sum) of the slave station number (STN) and all the data bytes in the packet. For polling packets, the BCC is simply the 2’s complement of STN. The BCC does not include any other message packets codes or response codes.

Figure 4.14
Formats for Half-Duplex Protocol



a) Polling Packet

b) Slave Message Link Packet

c) Master Message Link Packet

For example, if the master station wanted to send the data codes 8, 9, 6, 0, 2, 4, and 3 to slave station 20 hex (40 octal), the master message codes would be (in hex):

```
10  01  20  10  02  08 09 06 00 02 04 03   10   03  C0

DLE SOH STN DLE STX           Data        DLE ETX BCC
```

The sum of the STN and data bytes in this message packet is 40 hex. The BCC is the 2's complement of this sum, or C0 hex. This is shown in the following binary calculation:

```
0100 0000   hex
1011 1111   1s complement
      X+1
1100 0000   2s complement (E0 hex)
```

To transmit the STN or data value 10 hex, you must use the data code DLE DLE. However, only one of these DLE text characters is included in the BCC sum. For example, to transmit the values 8, 9, 6, 0, 10, 4, and 3 hex, a slave station would use the following message codes:

```
        Represents single text value of 10

10   02             08 09 06 00 10 10 04 03            10     03    D2
```

In this case, the sum of the data bytes is 2E hex because only one DLE text code is included in the BCC. So the BCC is D2 hex.

The BCC algorithm provides a medium level of data security. It cannot detect transposition by bytes during transmission of a packet. It also cannot detect the insertion or deletion of data values of zero within a packet.

Cyclic Redundancy Check

Calculate the CRC value on the value of the data bytes and the ETX byte (using the polynomial $x^{16} + x^{15} + x^2 + x^0$). To transmit the data value of 10 hex, you must use the data code DLE DLE. However, only one of these DLE data bytes is included in the CRC value. Embedded responses are not included in the CRC value.

At the start of a message packet, the transmitter clears a 16-bit register for the CRC value. As a byte is transmitted, it is exclusive-OR'd (with bit 0 to the right) to the right eight bits of the register. The register is then shifted right eight times with 0s inserted on the left. Each time a 1 is shifted out on the right, the following binary number is exclusive-OR'd with the 16-bit register value:

```
1010 0000 0000 0001
```

As each additional byte is transmitted, it is included in the value in the register the same way. After the ETX value is included in the value in the register and is transmitted, the value in the register is transmitted (right bit first) as the CRC field.

The receiver also calculates the CRC value and compares it to the received CRC value to verify the accuracy of the data received.

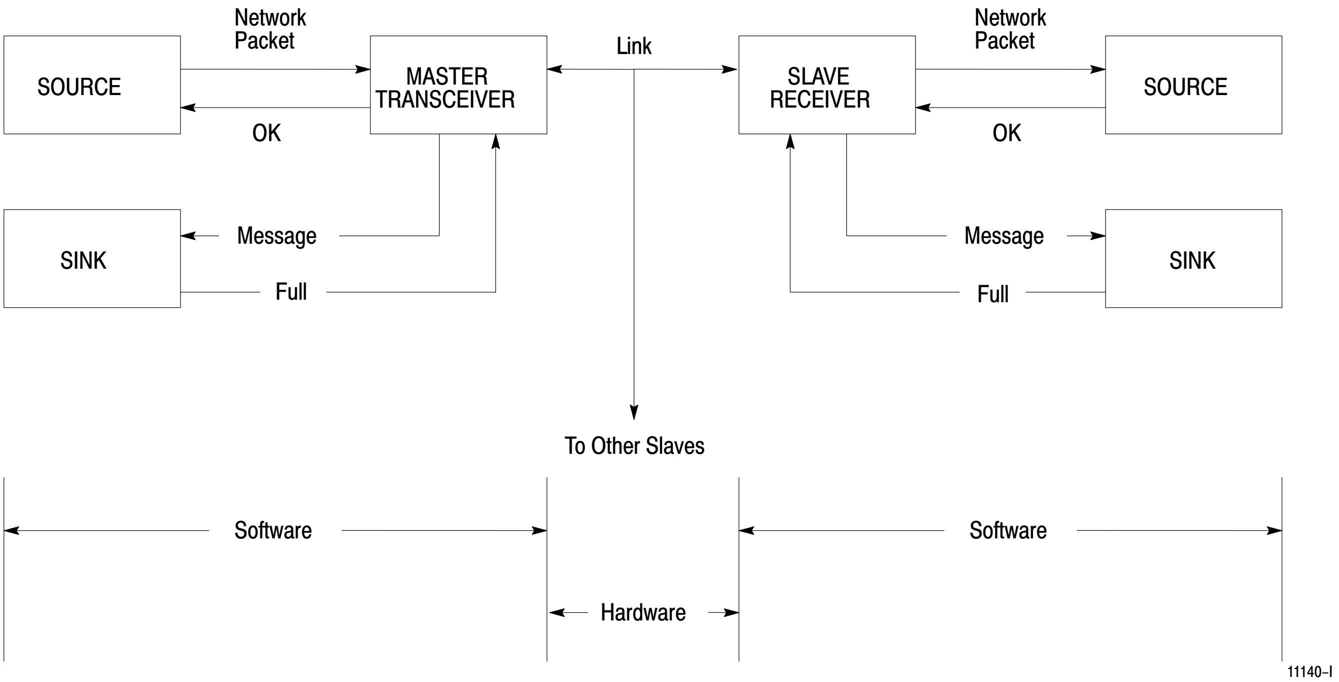
NOTE: In half duplex mode there is a BCC on the polling packet regardless of whether BCC or CRC error checking is used on messages.

Protocol Environment Definition

In each station there is a program connected to the link hardware that is called the transceiver. The master station has a more complex transceiver than the slaves, since it must include a polling algorithm. Only the slave's transceiver is defined here, as illustrated in Figure 4.15.

To fully define the environment of the protocol, the transceiver needs to know where to get the messages it sends and must have a means of disposing of messages it receives. These are implementation-dependent functions that shall respectively be called the message source and the message sink. We assume that the message source will supply one message at a time upon request from the transceiver, and will require notification of the success or failure of transfer before supplying the next message. Whenever the transceiver has received a message successfully, it will attempt to give it to the message sink. The possibility exists that the message sink will be full. The transceiver requires an indication of this.

Figure 4.15
Slave Transceiver



Message Characteristics

Ideally, the link protocol should not be at all concerned with the content or form of the messages it is transferring. However, half-duplex protocol places the following restrictions on the messages that are submitted to it for transfer:

1. Minimum size of a valid message is 6 bytes, maximum is 250 bytes.

2. Some protocol implementations require that the first byte of a message match the station address. These receivers will ignore messages that do not contain the correct address. This filtering is not required, since the message sink can also provide the address filtering function.

3. As part of the duplicate message detection algorithm, the receiver checks the second, third, fifth, and sixth bytes of each message. There must be a difference in at least one of these bytes between a message and its predecessor for it to be recognized as distinct rather than a retransmission of the previous message.

Master Polling Responsibilities

The master polling algorithm may vary depending on the expected flow of traffic through the system.

A simple master that does not expect unsolicited messages continuously polls each slave once in a round-robin fashion. If a message is received it should be handled, then the next station polled. An alternative scheme would poll each station repeatedly until it was empty, and then proceed to the next. Relaying of peer-to-peer messages is best left to the network layer, which is the lowest level of the body of software that includes the message source and sink.

Figure 4.16 is a flowchart of the software logic for implementing half-duplex protocol from the master station's point of view.

Slave Transceiver Actions

Since the transceiver receives "dirty" input from the physical world, it must be capable of responding to many adverse situations. Some of the things that could conceivably happen are listed below:

1. The message sink can be full, leaving nowhere to put a message.

2. A message can contain a parity error.

3. The BCC can be invalid.

4. The DLE SOH, DLE STX or DLE ETX BCC my be missing.

5. The message can be too long or too short.

6. A spurious control or data code can occur outside a message.

7. A spurious control code can occur inside a message.

8. The DLE ACK response can be lost, causing the transmitter to send a duplicate copy of a message that has already been passed to the message sink.

The slave is always in a passive mode until it receives a message. While waiting for a message, anything besides the DLE SOH or DLE ENQ is ignored. Note that in a single circuit system the slaves must be able to safely ignore everything sent by other slaves.

If a DLE SOH is received the BCC and the message buffer are reset. The next code received must be a data code and must equal the station address or 255 (if the station can receive broadcast messages). If there is no match, the station ignores the rest of the message and continues waiting for the start of the message.

If the station address matches, it is added to the BCC. The next code is received and must match DLE STX. If it doesn't, the station ignores the rest of the message. Otherwise it starts building a message.

While building a message, all data codes are stored in the message buffer and added to the BCC. If the buffer overflows, the receiver continues summing the BCC, but the data is discarded. If an error is detected, it is recorded. If any control code other than a DLE ETX BCC is received, the error flag, the BCC, the message size and the address (optionally) are all checked. If any of the tests fail, the message is ignored.

If the message is OK, its header is compared to the last message. If it is the same, the message is discarded and a DLE ACK is transmitted. If it is different, the new header is saved and the message is sent to the message sink. If the message can be stored, a DLE ACK is sent. If the message sink is full, the message is discarded and not acknowledged.

If while waiting for a message a DLE ENQ is received, the transceiver accepts the next two characters. The last character is read directly from the link, since it is a BCC and is not byte stuffed. If the station address does not match, or there is an error, the poll is ignored. If the poll is accepted, there are three possible situations.

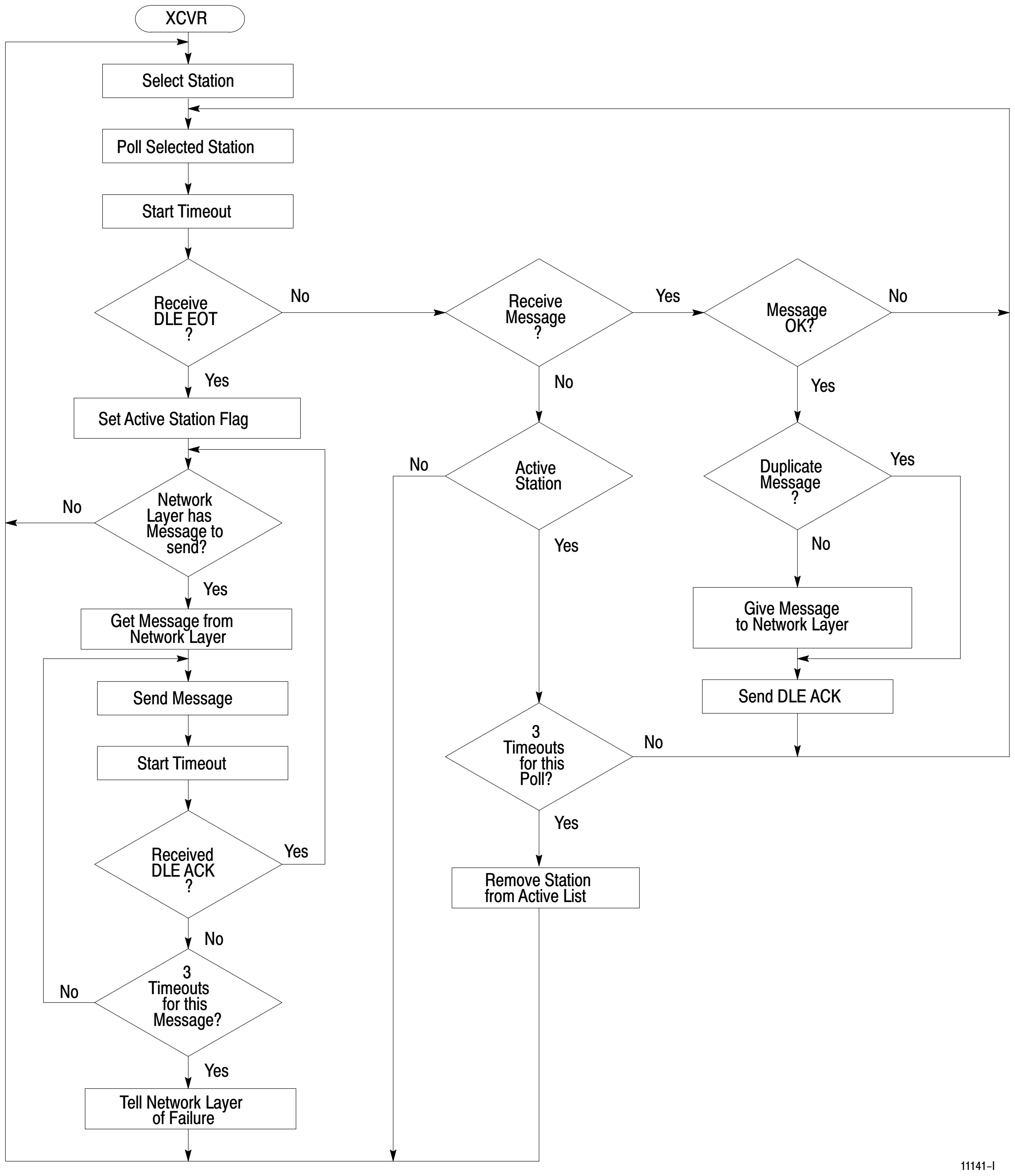
1. The transceiver could still be holding a message that it had transmitted previously, but had not been ACKed. There is a limit on the number of times each message can be sent. If this limit is exceeded when the poll is received, the message is returned to the message source with an error indication, and the transceiver tries to send the next message from the message source. If the limit is not exceeded the response to the poll will be to re-send the current message.

2. If no message is currently being held the transceiver tries to get one from the message source. If one is available the transceiver will initialize its re-try counter and transmit it in response to the poll.

3. If no message is available the response to a poll will be to transmit a DLE EOT.

When a message is transmitted after receiving a poll, its format is identical to a full-duplex message packet. After sending a message, the transceiver will hold the message until a DLE ACK is received, or the number of times the message has been polled exceeds the limit.

Figure 4.16
Implementation of Half-Duplex Protocol



When a DLE ACK is received, the message currently held is discarded. When the next poll is received the next message available from the message source is sent (or a DLE EOT).

When a DLE NAK is received, the transceiver takes messages from the source until it is empty. Each message will be discarded with an error code sent back to the message source. This can be used by the master to clear up the message source buffers of all slaves after the master has been down.

Half-Duplex Protocol Diagrams

The following figures show some of the events that occur on various interfaces. Time is represented as increasing from the top of the figure to the bottom.

Figure 4.17
Normal Message Transfer

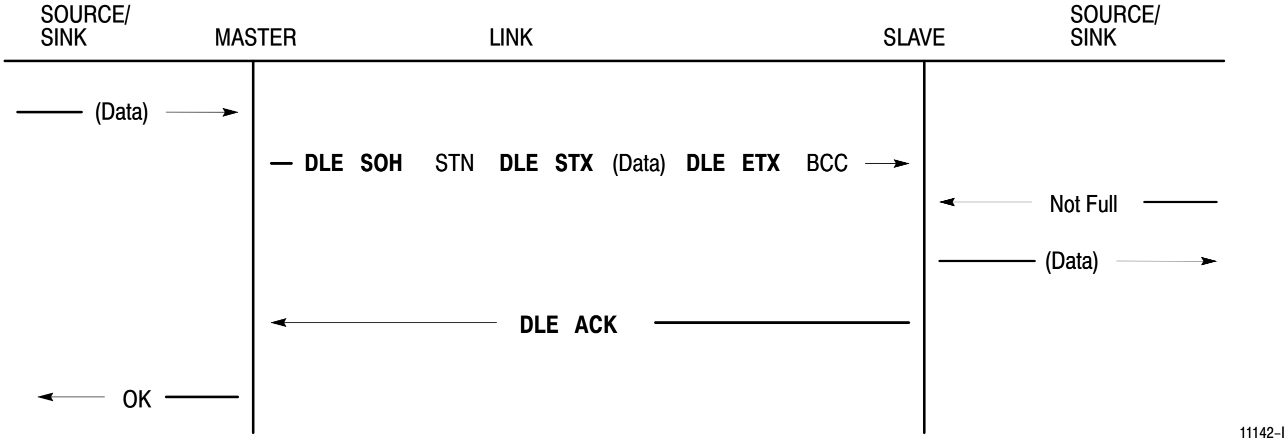


Figure 4.18
Message Transfer with Invalid BCC

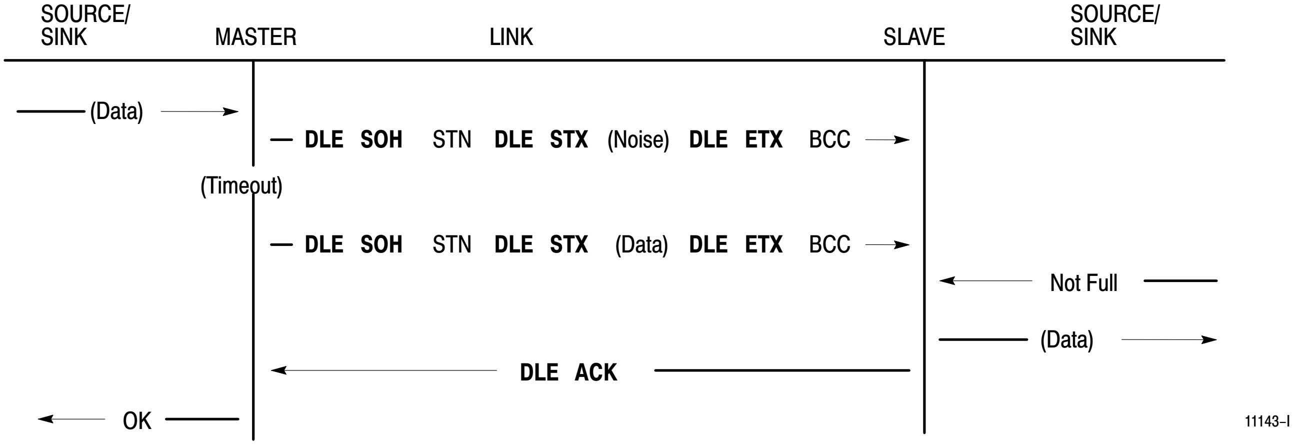


Figure 4.19
Message Transfer with ACK Destroyed

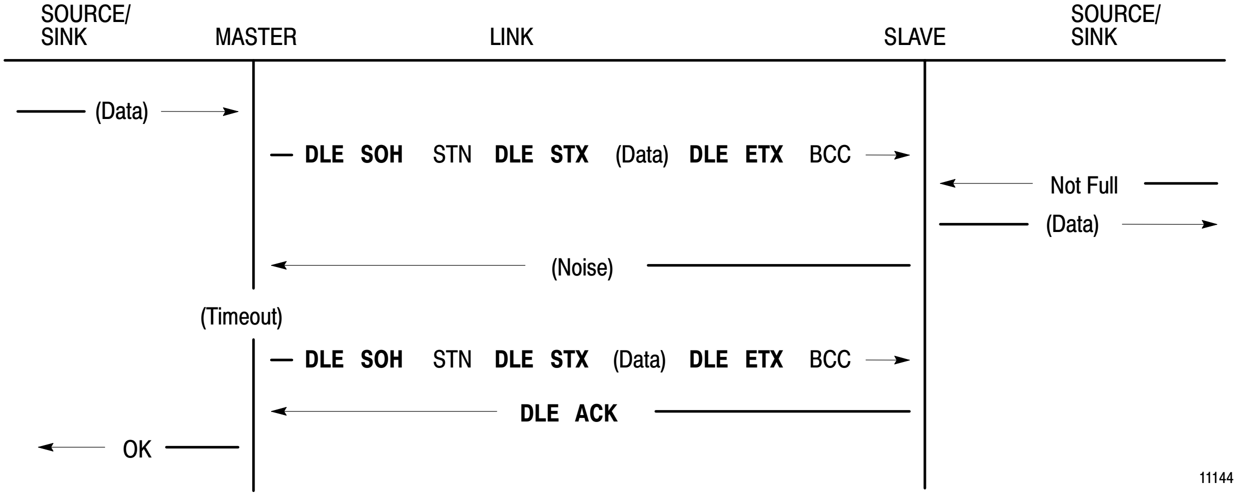


Figure 4.20
Poll with No Message Available

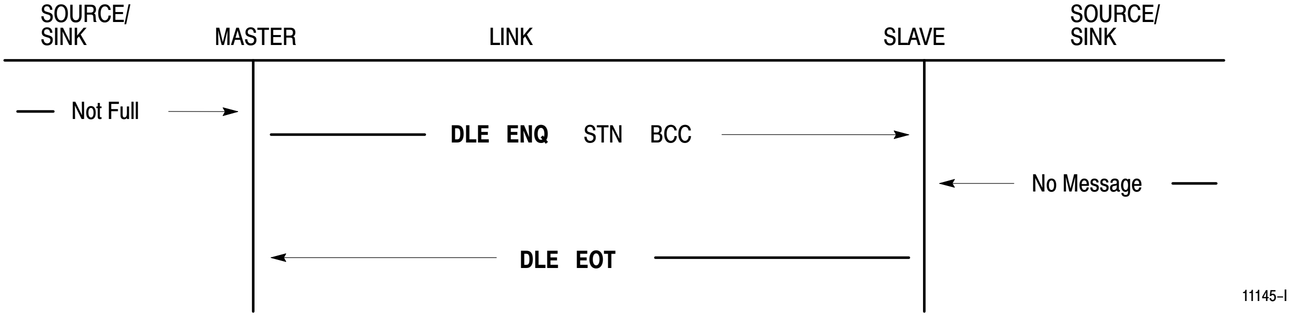
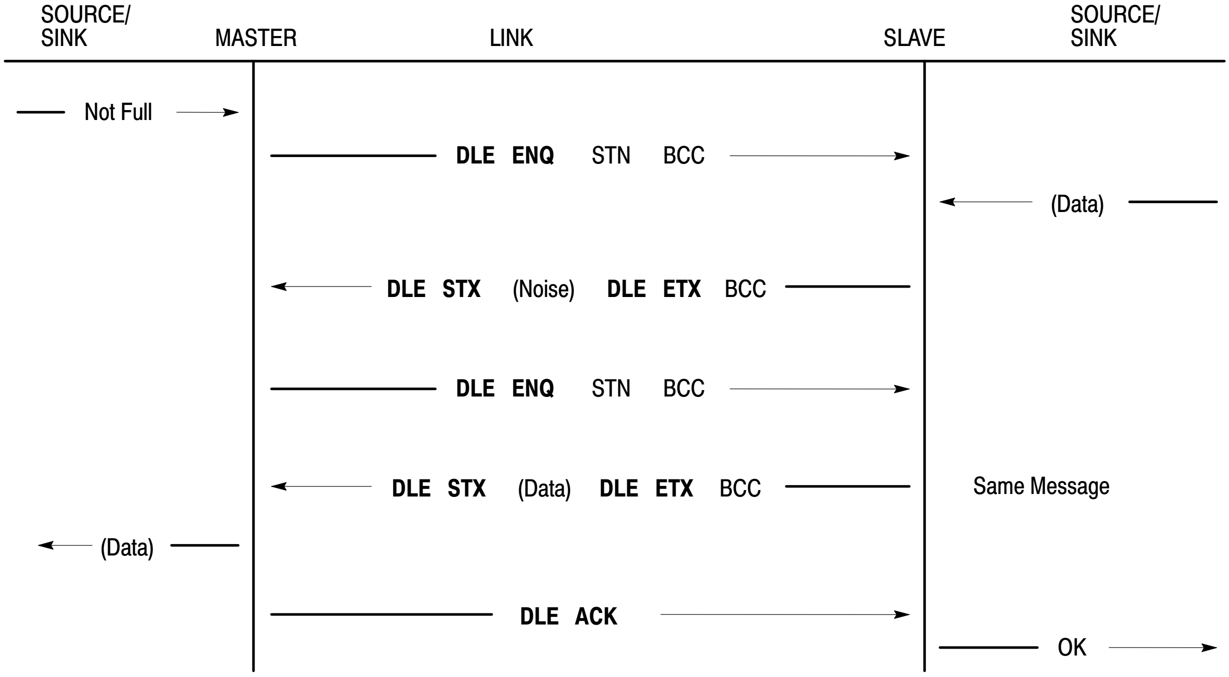


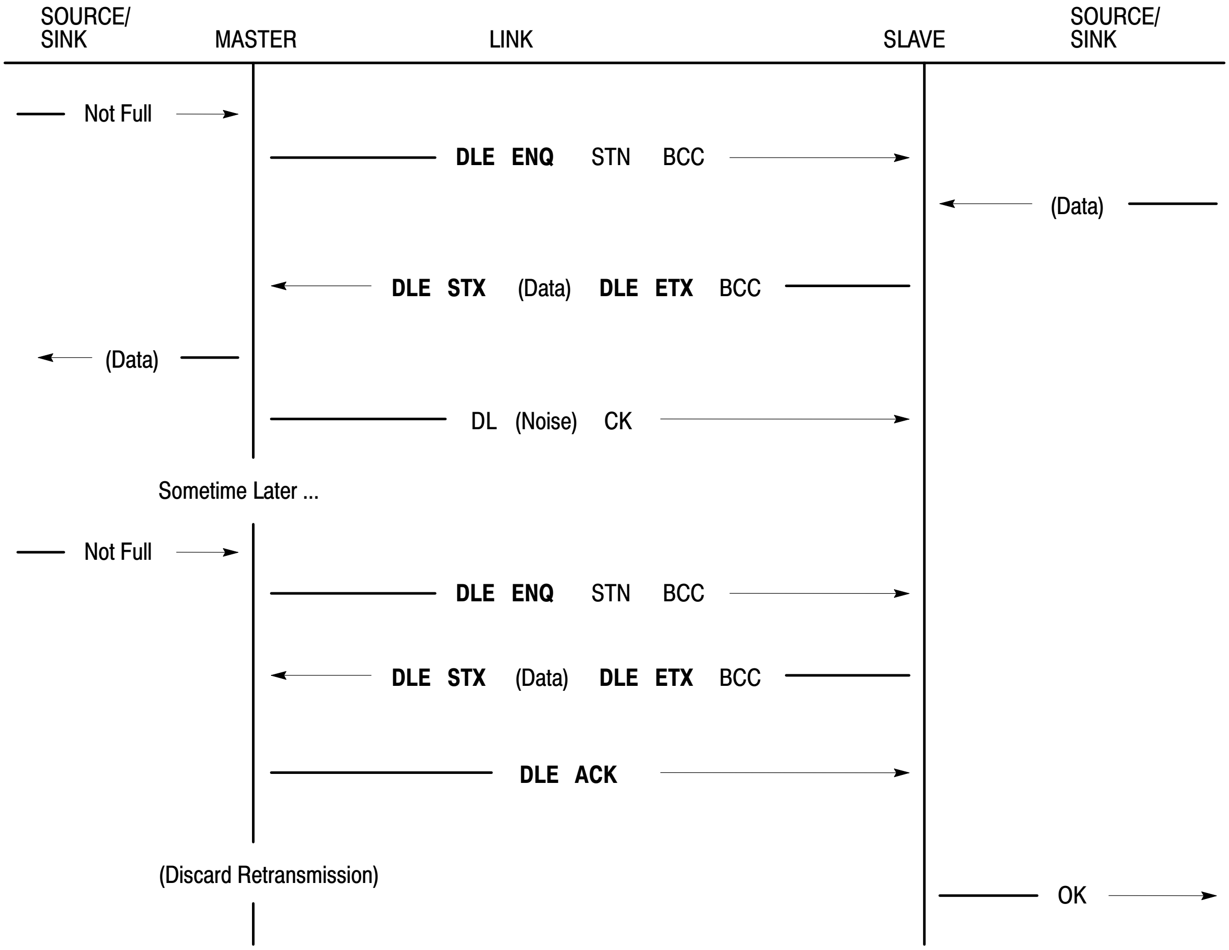
Figure 4.21
Poll with Message Returned



Note that the message sent for the second poll is the same as the first, since the first transmission was not ACKed.

11146-I

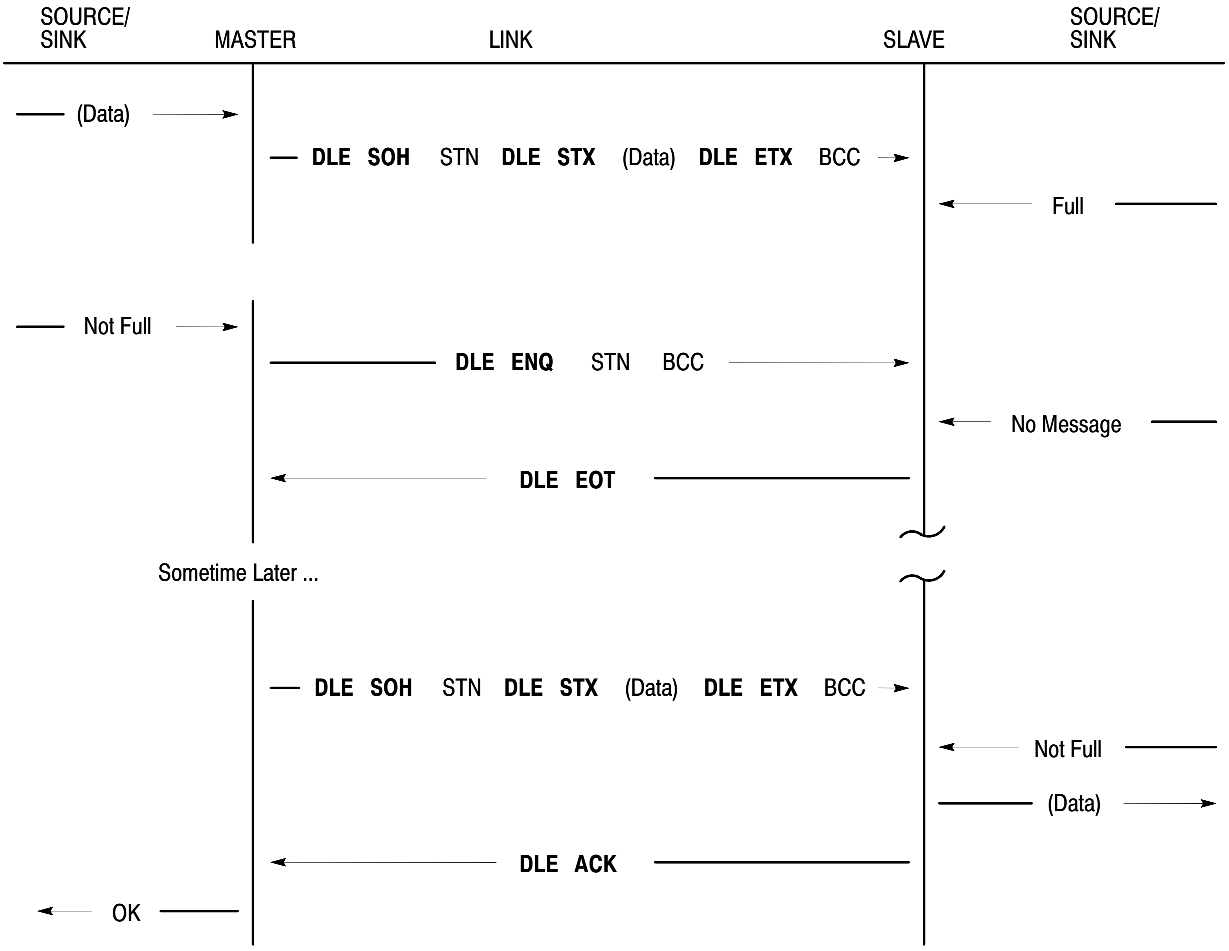
Figure 4.22
Duplicate Message Transmission



Note that the message sent for the second poll is the same as the first, since the first transmission was not ACKed. Also, the master should ACK and discard the second message, since it was a retransmission. This means that the master must either poll a station repeatedly until it receives a DLE EOT, without polling any other stations, to be sure it has detected all retransmissions. If each station is polled only once per cycle, the master must keep a record of the last header from each station, since other stations may transfer messages between retransmissions from a station.

11147–I

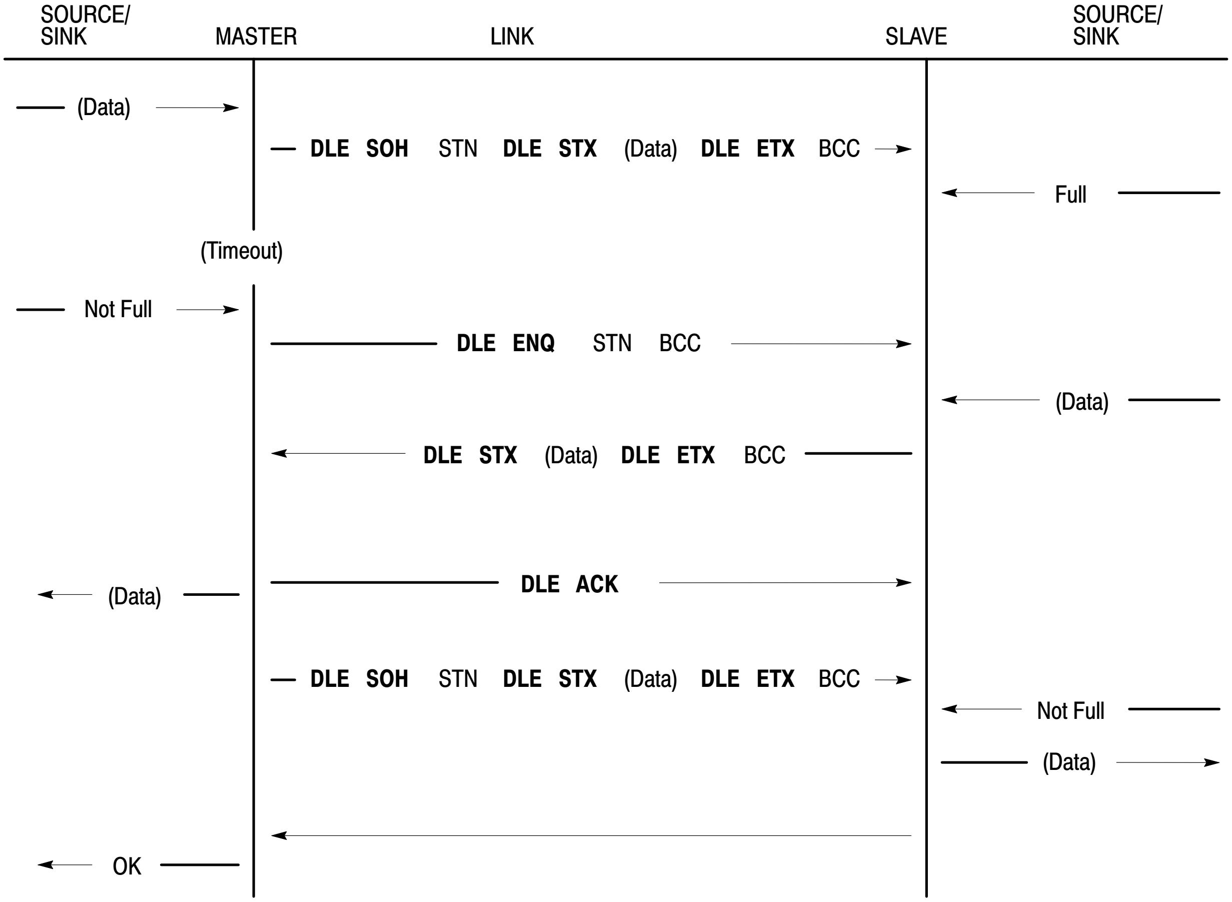
Figure 4.23
Message Sink Full, Case 1



When a station times out it is a good idea to poll it to see if it is there. If it answers with a DLE EOT to a message, but consistently fails to ACK a message, its message sink is probably full. If it responds DLE EOT to a poll, you must wait for the buffers to clear.

11148-I

Figure 4.24
Message Sink Full, Case 2



When a station's message source and sink share a commom memory pool, it may be that the message sink "full" indication results from an abundance of messages in the message source, which use up all free pool memory. In this case, the memory can be freed up by receiving messages from the station. Waiting for the memory to clear by the action of the station alone may not work, since it could be that the only way to free up space is to send a message to the master.

Line Monitoring

When monitoring half-duplex protocol in two-wire mode, you need to monitor only one line. The example below shows a message sent by the master and a reply sent by the slave in response to a poll. Slave responses are underlined.

Message from master to slave:

DLE SOH stn DLE STX message DLE ETX BCC **DLE ACK**

Message sent from slave to master in response to poll:

DLE ENQ stn BCC **DLE STX message DLE ETX BCC**
DLE ACK

Poll with a DLE EOT in response:

DLE ENQ stn BCC **DLE EOT**

Chapter 5

Message Packet Formats

General

This chapter describes how your computer application programs should format command and reply messages for transmission over the RS-232-C link. It covers both application level and network level functions. Basically, the application layer specifies the contents of messages and initiates their transmissions. The network layer takes the information from the application layer and formats it in a way that is acceptable to the data link layer described in chapter 4.

Application Layer

Application programs are of two types: command initiators and command executors. This corresponds with the two message types:

- Command messages - sent by command initiators to command executors
- Reply messages - sent by command executors in response to command messages received from command initiators

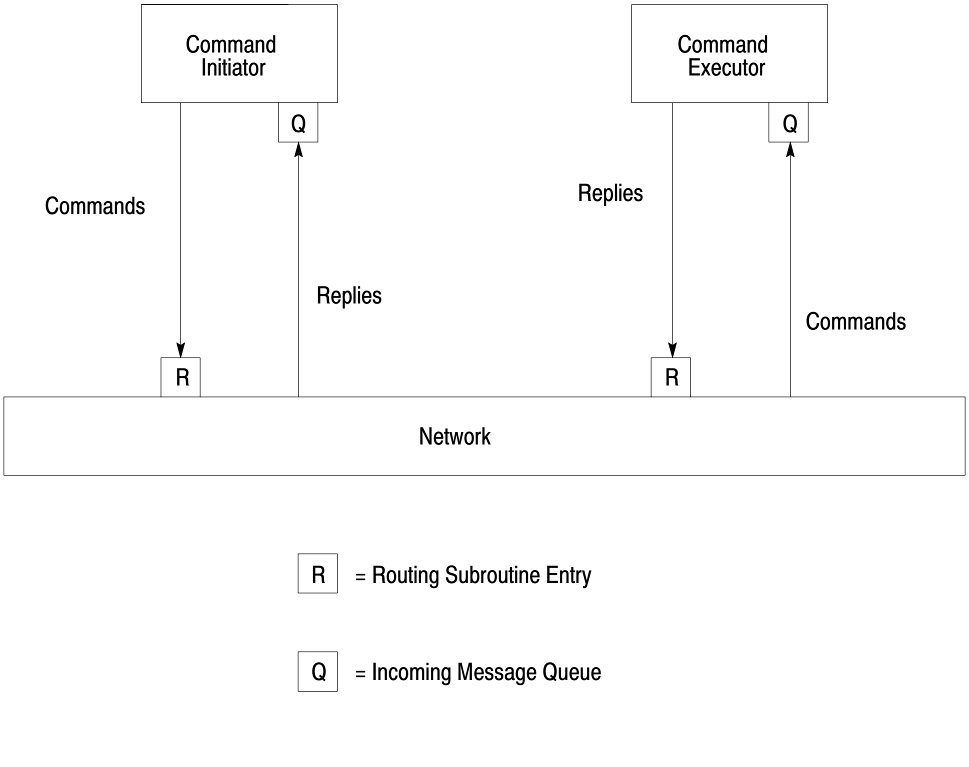
Each command message requires one reply message.

Command initiators specify which command function to execute at a particular destination station. The command executor at that destination station is responsible for interpreting the command message and executing the specified command function. The command executor also issues a reply message for each command it receives. If it cannot execute the received command, the command executor must generate the appropriate error message.

Network Layer

Internally, the KE/KF module uses a routing subroutine and an message queue to implement the network layer. When the module receives a message over its RS-232-C port, it puts that message in the queue. The routing subroutine then takes the message from the queue and transmits it over the Data Highway link. The module also queues messages received from the Data Highway, and the routing subroutine retransmits those messages over the RS-232-C link. Figure 5.1 illustrates this model.

Figure 5.1
Application Model



Messages do not necessarily arrive at their destination in the same order in which they were sent. It is impossible for the network layer to guarantee delivery, and in some cases it may not be possible to provide notification of non-delivery. In particular, the network layer does not notify the command executor about non-delivery of a reply message. Therefore, it is advisable that your command initiator maintain a timer for each outstanding command message. If the time limit expires before the command initiator receives the corresponding reply to its command, it can either re-transmit the same command message or indicate an error condition.

If your network layer software cannot deliver a command message, it should generate a reply message with an error code in it and send that reply to the command initiator. If it cannot deliver a reply message, the network layer software should simply destroy the reply without notification to the command executor.

If your link layer software cannot deliver a message over the RS-232-C link, it also should return an error message to the command initiator.

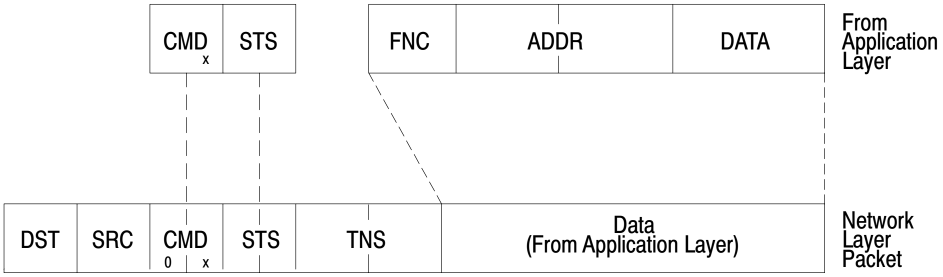
Message Packet

The message bytes contain unsigned binary data from both the application layer and the network layer. Figure 5.2 shows the general format of a command message. Network layer fields are shaded. The meanings of the bytes are:

- DST - destination station for the message
- SRC - source station of the message
- CMD - command code
- STS - status code
- TNS - transaction
- RNG - rung number for PC command message
- SQN - sequence number of message
- FNC - function code
- ADDR - address of memory location
- DATA - data values being transferred by the message

These bytes are described in more detail below. Not all command messages have FNC, ADDR, or DATA bytes.

Figure 5.2
Command Message Packet Format

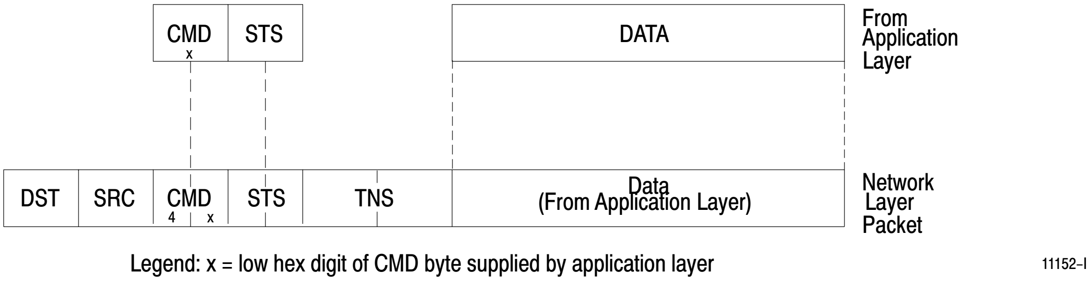


Legend: x = low hex digit of CMD byte supplied by application layer

11151–I

Figure 5.3 shows the general format of a reply message. Network layer fields are shaded. The definitions of these bytes are the same as for command messages. Not all reply messages have DATA bytes.

Figure 5.3
Reply Message Packet Format



Legend: x = low hex digit of CMD byte supplied by application layer

Note that the bytes are shown from left to right in the order in which they are transmitted across the link.

DST and SRC

The DST (destination) byte contains the station number of the station that is the ultimate destination of the message. The SRC (source) byte is the station number of the station that originated the message.

The network layer supplies the DST and SRC values. Allowed values for these bytes are 0 to 254 decimal.

Note that you can form the DST and SRC bytes of a reply message simply by interchanging the DST and SRC bytes of the corresponding command message.

CMD and FNC

The CMD (command) and FNC (function) bytes together define the activity to be performed by the command message at the destination station. CMD defines the command type and FNC, if used, defines the specific function under that command type.

Appendix A lists all the available CMD and FNC codes in hexadecimal notation. The exact format for a particular message depends on the CMD and FNC values. Section titled “Message Packet” explains the use of each command or function and gives a detailed description of the message text format for each.

Figure 5.4 shows the format for the CMD byte itself. Bits 0 through 4 contain the command code. Bits 4 and 7 should always be set to 0. Bit 5 is the priority indicator; set it to 0 for normal priority messages and 1 for high priority messages (chapter 2). Bit 6 is the command/reply indicator; set it to 0 in a command message and 1 in a reply message.

Note that reply messages also contain a CMD byte. To form the CMD value for a reply, the network layer copies the CMD value from the corresponding command message and sets the reply bit (bit 6) to 1.

Figure 5.4
CMD byte Format

Bit	7	6	5	4	3	2	1	0
	0	r	p	0	command			

11153–I

The application layer supplies the FNC value and the command code (bits 0 to 3) of the CMD byte for each command message. The network layer supplies bits 4 through 7 of the CMD byte.

STS

The STS (status) byte indicates the status of the message transmission. In command messages, the application program should always set the STS value to 0. In reply messages, the STS byte may contain one of the status codes listed in chapter 7. Note that an STS value of 0 in a reply message means “no error.”

In a reply message, the STS byte is divided between application layer and network layer. The network layer uses bits 0 through 3 of the STS byte to report local errors (those errors that occur when the network layer attempts to transmit a message across the link). The application layer uses bits 4 through 7 of the STS byte to report remote errors (those errors that occur when the command executor at the destination station tries to execute the command message). Chapter 7 explains local and remote error codes.

TNS

The TNS (transaction) bytes contain a unique 16-bit transaction identifier. A complete transaction consists of a command message transmitted by a PC station. RNG contains the number of the PC program rung that generated the command, and SQN contains the transmission sequence number. For command messages transmitted by your computer station, your application programs can use the RNG and SQN bytes together to store a single 16-bit transaction number.

For command messages transmitted by a PC station, the station interface module assigns the TNS values. For each command message transmitted by your computer station, your network level software must assign a unique 16-bit transaction number and a simple way to generate this number is to maintain a 16-bit counter at the network layer. Increment the counter every time your command initiator (application program) creates a new message, and store the counter value in the two TNS bytes of the new message.

When the command initiator receives a reply to one of its command messages, it can use the RNG and SQN bytes to tie the reply message to its corresponding command. If the RNG and SQN bytes of a reply message match the RNG and SQN bytes of a command message, then that reply is the appropriate one for that command.

Whenever your command executor receives a command from another station, it should copy the RNG and SQN fields of the command message into the same fields of the corresponding reply message. Do not change the RNG and SQN values in a reply message. If you do, the command initiator will not be able to match its command to the corresponding reply message.

Note that the low byte (least significant bits) of your TNS value will be transmitted across the link before the high byte (most significant bits).

At any instant, the combination of SRC, CMD, RNG, and SQN are sufficient to uniquely identify every message packet in transit. At least one of these fields in the current message must be different than the corresponding field in the last message received by a command executor. If none of these fields is different, the command executor ignores the current received message. This process is called duplicate message detection.

ADDR

The ADDR (address) field is a 2-byte field that contains the address of a memory location in the command executor. ADDR specifies the address where the command is to begin executing. For example, if the command is to read data from the command executor, ADDR specifies the address of the first byte of data to be read.

The first byte of the ADDR field contains the low (least significant) byte of the address, and the second byte of ADDR contains the high byte of the address. Chapter 6 gives more details on the contents of the ADDR field.

Note that the ADDR field specifies a byte address, not a word address as in PC programming. Chapter 6 explains how to convert PC word addresses to byte addresses.

SIZE

The SIZE byte specifies the number of data bytes to be transferred by a message. This field appears in read commands, where it specifies the number of data bytes that the responding station must return in its reply message. The allowed value for SIZE will vary with the type of command, as indicated in the reply formats below.

DATA

The DATA field contains binary data from the application program. The number of data bytes in a message depends on the command or function being executed, as indicated in section titled “Message Packet.” Chapter 6 gives some details on the format for the data field.

Message Formats

This section presents the detailed message formats for each type of command and reply message. For this presentation, the command message formats are discussed in the following order:

If you want this command:	use this command code:	use this function code:
Basic command set		
Diagnostic counters reset	06	07
Diagnostic loop	06	00
Diagnostic read	06	01
Diagnostic status	06	03
Protected bit write	02	N/A
Protected write	00	N/A
Set ENQs	06	06
Set NAKs	06	05
Set timeout	06	04
set variables	06	02
Unprotected bit write	05	N/A
Unprotected read	01	N/A
Unprotected write	08	N/A
PLC commands		
Disable outputs	07	00
Enable program	07	01
Enable scan	07	03
Physical read	04	N/A
Physical write	03	N/A
PLC-2 commands		
Enter download mode	07	04

If you want this command:	use this command code:	use this function code:
Enter upload mode	07	06
Exit download/upload mode	07	05
Physical read	04	N/A
Physical write	03	N/A
Set data table size	06	08
PLC-3 commands		
Bit writes	OF	02
Download request	OF	05
File read	OF	04
File write	OF	03
Physical read	OF	08
Physical write	OF	08
Restart request	OF	0A
Shutdown request	OF	07
Upload request	OF	06
Word range read	OF	01
Word range write	OF	00
PLC-4 commands		
Allocate	OE	05
Deallocate	OE	06
Initialize processor	OE	0C
Physical read	OE	0D
Physical write	OE	0E
Physical write with mask	OE	0F
Set to program mode	OE	01
Set to run mode	OE	02
Set to single step test mode	OE	04
Set to test mode	OE	03

Basic Command Set

Basic commands include those that can generally be executed by any PC station on the communication link, regardless of the type of PC controller at that station. In some cases, switch settings on the station interface module can disable execution of a particular type of command at that station. For more details, refer to the user’s manual for the station interface module.

Basic commands are in two categories:

Privileged Commands [1]

- Diagnostic counters reset
- Diagnostic loop
- Diagnostic read
- Diagnostic status
- Set ENQs
- Set NAKs
- Set timeout
- Set variables

Non-privileged Commands[2]

- Protected bit write
- Protected write
- Unprotected bit write
- Unprotected read
- Unprotected write

[1] Computer initiates commands and PCs execute command

[2] Computer and PC stations can initiate commands; only PCs can execute commands (unless the computer is programmed for execution).

Diagnostic Counters Reset

This command resets to zero all the diagnostic timers and counters in the station interface module. The diagnostic status command gives the starting address for this block of counters and timers.

Command Format:

DST	SRC	CMD	STS	TNS	FNC
		46			07

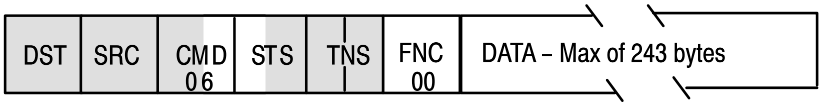
Reply Format:

DST	SRC	CMD	STS	TNS
		46		

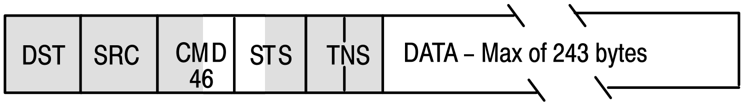
Diagnostic Loop

You can use this command to check the integrity of transmissions over the communication link. The command message transmits up to 243 bytes of data to a station interface module. The receiving module should reply to this command by transmitting the same data back to the originating station.

Command Format:



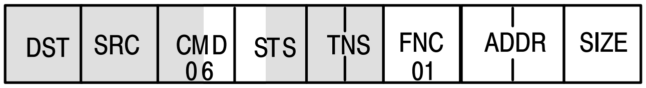
Reply Format:



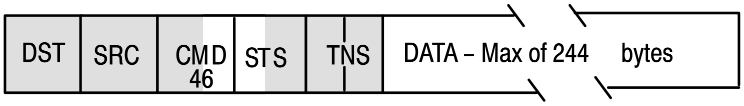
Diagnostic Read

This command reads up to 244 bytes of data from the PROM or RAM of the station interface module. You can use it to read the module's diagnostic timers and counters. Use the diagnostic status command to obtain the starting address of the diagnostic counters:

Command Format:



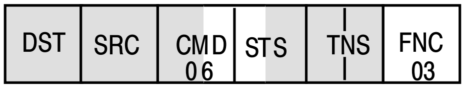
Reply Format:



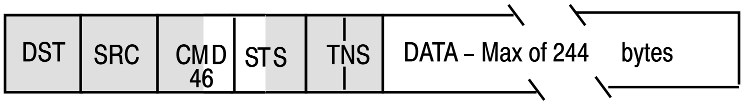
Diagnostic Status

This command reads a block of status information from the station interface module. The reply to this command contains the status information in its DATA field.

Command Format:



Reply Format:



The status information varies with the type of station interface module. Table 5.A describes the contents of the status DATA field for 1771-KA, 1771-KC/KD, 1771-KE/KF, 1771-KG and 1174-KA modules. Table 5.B lists the status DATA for 1773-KA module. Table 5.C describes this DATA for 1775-KA modules.

Table 5.A
Contents of Status DATA for 1771-KA, 1771-KC/KD, 1771-KE/KF, 1771-KG, and 1774-KA Modules

Byte	Meaning	
1	Operating status of PC processor:	
	Bits 0 to 2:	0 = Program load mode
		1 = Test mode
		2 = Run mode
		3 = (not used)
		4 = Remote program load
		5 = Remote test
		6 = Remote run monitor (PLC)
	Bit 3	0 = Normal
		1 = No communication with PC
	Bit 4:	0 = Normal
		1 = Download mode
	Bit 5:	0 = Normal
		1 = Format error in communication zone of PC program
	Bits 6 and 7:	Always zero
2	Type of station interface module and processor:	
	Bits 0 to 3	0 = 1771-KC/KD module
		1 = 1771-KA module also KA2
		2 = 1774-KA module
		3 = 1771-KE/KF module
		4 = 1771-KG module
		5 = (not used)
		6 = 1775-KA, data highway port
		7 = 1775-KA, RS-232-C port
		8 = 1771-KA, data highway port
		9 = 1773-KA, RS-232-C port
	Bits 4 to 7	0 = PLC processor
		1 = PLC-2 processor
		2 = PLC-2/20 (LP1) processor
		3 = Mini-PLC-2 processor
		4 = PLC-3 processor
		5 = PLC-2/20 (LP2) processor
		6 = PLC-2/15 processor
		7 = PLC-2/30 processor

Byte	Meaning	
		8 = PLC-4 Microtrol processor
		15= Computer
3, 4	Octal word address of the start of PC program	
5, 6	Memory size (number of bytes) for PLC processors; zero otherwise	
7, 8	Starting byte address of diagnostic counters and timers	
9	Series and revision level of station interface module:	
	Bits 0 to 4:	0 = Revision A
		1 = Revision B
		etc.
	Bits 5 to 7:	0 = Series A
		1 = Series B
		etc.
		(KA 2 series B, revision A)
10	Settings of the option switches on the station interface module. This byte is not used in replies by 1771-KC/KD or 1771-KE/KF modules. For the other modules, the bits of this byte are defined as follows:	
	1771-KA Module	
	Bits 0 to 1:	0 = 56,700 bits per second
		1 = 76,800 bits per second
		2 = 38,400 bits per second
		3 = 115,200 bits per second
	Bit 2:	0 = All other PLC-2 Family processors
		1 = PLC-2 processor
	Bit 3:	0 = Protected commands enabled
		1 = Protected commands disabled
	Bit 4:	0 = Unprotected commands enabled
		1 = Unprotected commands disabled
	Bit 5:	Not used
	Bit 6:	0 = Physical writes enabled
		1 = Physical writes disabled
	Bit 7:	0 = Transmission of unprotected commands enabled
		1 = Transmission of unprotected commands disabled
	1771-KG Module	
Byte	Meaning	

Byte	Meaning	
	Bit 0:	0 = Half-duplex protocol
		1 = Full-duplex protocol
	Bit 1:	0 = Physical writes enabled
		1 = Physical writes disabled
	Bit 2:	0 = Unprotected commands enabled
		1 = Unprotected commands disabled
	Bit 3:	0 = Embedded responses enabled
		1 = Embedded responses disabled
	Bit 4:	0 = Even parity
		1 = No parity
	Bits 5 to 7:	0 = 19,200 bits per second
		1 = 9600 bits per second
		2 = 4800 bits per second
		3 = 2400 bits per second
		4 = 1200 bits per second
		5 = 600 bits per second
		6 = 300 bits per second
		7 = 110 bits per second
	1774-KA Module	
	Bit 0:	0 = Unprotected commands enabled
		1 = Unprotected commands disabled
	Bit 1:	Not used
	Bit 2:	0 = Physical writes enabled
		1 = Physical writes disabled
	Bit 3:	0 = Transmission of unprotected commands enabled
		1 = Transmission of unprotected commands disabled
	Bit 4:	0 = Data highway port B is connected
		1 = Data highway port A is connected
	Bit 5:	0 = PLC outputs held in last state
		1 = PLC outputs turned off
	Bit 6 to 7:	0 = 57,600 bits per second
		1 = 76,800 bits per second
		2 = 38,400 bits per second
		3 = 115,200 bits per second

Table 5.B
Contents of Status DATA for 1773-KA Modules

Byte	Meaning	
1	Operating status of controllers on the loop:	
	Bit 0 = 1 if controller #1 is active	
	Bit 1 = 1 if controller #2 is active	
	Bit 2 = 1 if controller #3 is active	
	Bit 3 = 1 if controller #4 is active	
	Bit 4 = 1 if controller #5 is active	
	Bit 5 - 1 if controller #6 is active	
	Bit 6 = 1 if controller #7 is active	
	Bit 7 = 1 if controller #8 is active	
2	Station interface type:	
	Bits 0 to 3:	8 = 1773-KA, data highway port
		9 = 1773-KA, RS-232-C port
	Bits 4 to 7:	8 = PLC-4 Microtrol processor
3, 4	Data Highway port options:	
	Bit 0:	0 = 57,600 bits per second
		1 = 38,400 bits per second
	Bit 1:	Not used
	Bit 2:	0 = Privileged commands enabled
		1 = Privileged commands disabled
	Bit 3:	0 = Unprotected commands enabled
		1 = Unprotected commands disabled
	Bit 4:	0 = Protected commands enabled
		1 = Protected command disabled
	Bit 5 to 7: Not used	
	Bit 8 to 15:	Octal station number
5, 6	RS-232-C port options:	
	Bit 0:	0 = Even parity
		1 = No parity
	Bit 1 to 3: 0 = 19,200 bits per second	
		1 = 9600 bits per second
		2 = 4800 bits per second
		3 = 2400 bits per second
		4 = 1200 bits per second
		5 = 600 bits per second
		6 = 300 bits per second
		7 = 110 bits per second
	Bits 4 to 10:	Not used
	Bit 11:	0 = Protected commands enabled

Byte	Meaning	
		1 = Protected commands disabled
	Bit 12:	0 = Embedded responses enabled
		1 = Embedded responses disabled
	Bit 13:	0 = Unprotected commands enabled
		1 = Unprotected commands disabled
	Bit 14:	0 = Privileged commands enabled
		1 = Privileged commands disabled
	Bit 15:	0 = Half-duplex protocol
		1 = Full-duplex protocol
7, 8	Starting byte address of diagnostic timers and counters	
9	Module series and revision level:	
	Bits 0 to 4:	0 = Revision A
		1 = Revision B etc.
	Bits 5 to 7:	0 = Series A
		1 = Series B etc.
10	Not used	
11 to 114	eight 13-byte groups of processor status data, one group for each of eight possible controllers on the loop. If a particular controller on the loop is not active or does not respond to the diagnostic status command, its 13 status bytes will all be zeroes. Otherwise, each group of processor status bytes will contain the following information: Byte 1: Program I.D. 2: Processor I.D. 3: Pointer to start of program 4: Pointer to end of available memory 5: Size of I/O 6: Processor error code 7: Error word address (low byte) 8: Error word address (high byte) 9: Processor mode 10: Pointer to END statement (low byte) 11: Pointer to END statement (high byte) 12: Pointer to end of used memory (low byte) 13: Pointer to end of used memory (high byte)	

Table 5.C
Contents of Status DATA for 1775-KA Modules

Byte	Meaning	
1	Operating status of PLC-3 processor:	
	Bits 0 to 1:	0 = Program mode
		1 = Test mode
		2 = Run mode
	Bit 2:	Not used
	Bit 3:	0 = Normal
		1 = Major processor fault
	Bit 4:	0 = Normal
		1 = Shutdown requested
	Bit 5:	0 = Normal
		1 = Shutdown in effect
	Bits 6 to 7:	Not used
2	Type of station interface:	
	Bits 0 to 3:	6 = 1775-KA, data highway port
		7 = 1775-KA, RS-232-C port
	Bits 4 to 7:	4 = PLC-3 processor
3	Current context (stored in bits 4 to 7)	
4	Thumbwheel number	
5, 6	Mode control word. The logical address of the mode control word is E0.0.0.8.	
7, 8	Starting byte address of the diagnostic counters and timers. There is a separate block of diagnostic timers and counters for the data highway port and the RS-232-C port. The address given here is the one for the port that received the diagnostic status command.	
9	Series and revision number of the 1775-KA module:	
	Bit 0 to 4:	0 = Revision A
		1 = Revision B
		etc.
	Bits 5 to 7:	0 = Series A
		1 = Series B
		etc.
10	Not used	
11 to 14	The physical address of the unused word of PLC-3 system memory. This is the physical address corresponding to the logical address E60.0.0.0.	
15 to 18	The total number of words in PLC-3 system memory (both used and unused). This is the physical word address corresponding to the logical address E63.0.0.0.	

Protected Bit Write

This command sets or resets individual bits within limited areas of the PC data table memory. Its access is limited by memory access rungs in the communication zone of the PC's ladder diagram program.

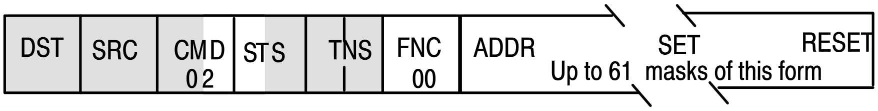
The data field in this packet consists of 4-byte blocks, each of which contains a 16-bit address field, a set mask, and a reset mask. Use the ADDR field to specify the address of the byte to be modified in the PC data table memory. Put the low byte (least significant bits) of the PC address value into the first byte of the ADDR field. Refer to chapter 6 for more details on how to specify an address value.

Use the SET mask to specify which bits to set to 1 in the addressed PC byte. A 1 in a bit position of the SET mask means to set the corresponding bit in the addressed PC byte to 1; a 0 in a bit position of the SET mask means to leave the corresponding bit in the PC byte unchanged.

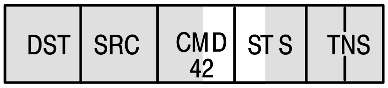
Use the RESET mask to specify which bits to reset to 0 in the addressed PC byte. A 1 in a bit position of the RESET mask means to reset the corresponding bit in the addressed PC byte to 0; a 0 in a bit position of the reset mask means to leave the corresponding bit in the PC byte unchanged.

Note that the interface module at the receiving PC station executes this command by first making a copy of the addressed PC byte. It then sets or resets the appropriate bits and writes the byte back into PC memory. At the same time, the PC processor can be changing the states of the original bits in memory. Because of this, some data bits may unintentionally be overwritten.

Command Format:



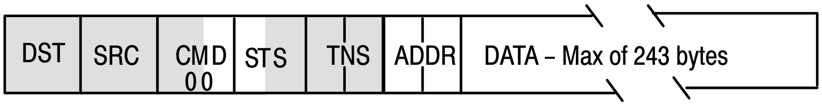
Reply Format:



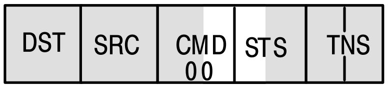
Protected Write

This command writes words of data into limited areas of the PC data table memory. Its access is limited by memory access rungs in the communication zone of the PC's ladder diagram program.

Command Format:



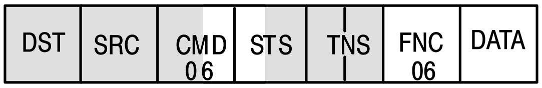
Reply Format:



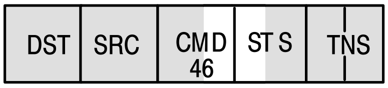
Set ENQs

This command sets the maximum number of ENQs that the station interface module will issue per message transmission. Put the number in the DATA field. The default setting for the KE/KF module is 10 ENQs per transmission.

Command Format:



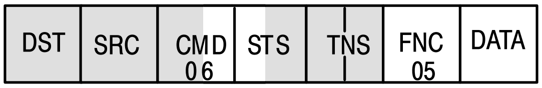
Reply Format:



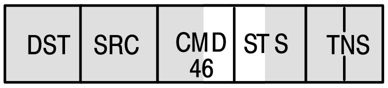
Set NAKs

This command sets the maximum number of NAKs that the station interface module will accept per message transmission. Put the number in the DATA field. The default setting for the KE/KF module is 3 NAKs per transmission.

Command Format:



Reply Format:



Set Timeout

This command sets the maximum amount of time that the station interface module will wait for an acknowledgement to its message transmission. The setting is expressed as the number of cycles of an internal clock, where 40 cycles equals 1 second. Put the number of desired cycles in the DATA field. The default setting for the KE/KF module is 128 cycles, or about 3 seconds.

Command Format:

DST	SRC	CMD	STS	TNS	FNC	DATA
		06			04	

Reply Format:

DST	SRC	CMD	STS	TNS
		46		

Set Variables

This command is a combination of the above three commands. It sets the maximum ENQs, NAKs, and timeout all at once. Put the timeout setting in the first byte of the DATA field, the NAKs setting in the second byte, and the ENQs in the third byte. If you do not specify a data value for any one the variables in this command, that variable is automatically reset to zero.

Command Format:

DST	SRC	CMD	STS	TNS	FNC	DATA – 3 bytes
		06			02	

Reply Format:

DST	SRC	CMD	STS	TNS
		46		

Unprotected Bit Write

This command sets or resets individual bits in any area of PC data table memory.

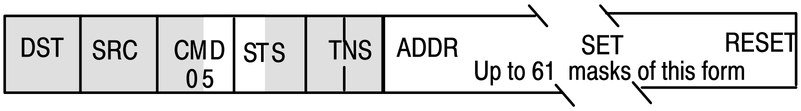
The data field in this packet consists of 4-byte blocks, each of which contains a 16 bit-address field, a set mask, and a reset mask. Use the ADDR field to specify the address of the byte to be modified in the PC data table memory. Put the low byte (least significant bits) of the PC address value into the first byte of the ADDR field. Refer to chapter 6 for more details on how to specify an address value.

Use the SET mask to specify which bits to set to 1 in the addressed PC byte. A 1 in a bit position of the SET mask means to set the corresponding bit in the addressed PC byte to 1; a 0 in a bit position of the SET mask means to leave the corresponding bit in the PC byte unchanged.

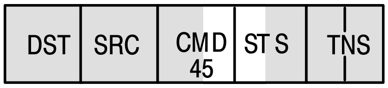
Use the RESET mask to specify which bits to reset to 0 in the addressed PC byte. A 1 in a bit position of the RESET mask means to reset the corresponding bit in the addressed PC byte to 0; a 0 in a bit position of the RESET mask means to leave the corresponding bit in the PC byte unchanged.

Note that the interface module at the receiving PC station executes this command by first making a copy ofle at the receiving PC station executes this command by first making a copy of the addressed PC byte. It then sets or resets the appropriate bits and writes the byte back into PC memory. At the same time, the PC processor can be changing the states of the original bits in memory. Because of this, some data bits may unintentionally be overwritten.

Command Format:



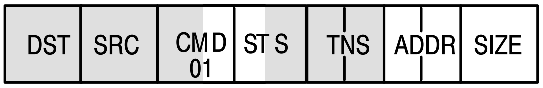
Reply Format:



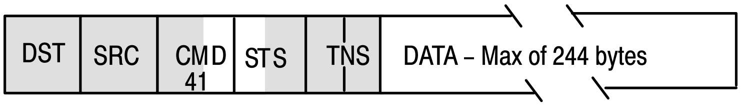
Unprotected Read

This command reads words of data from any area of PC data table memory. Use the SIZE field to specify the number of bytes to be read. To specify a number of PC words, SIZE should be an even value because PC words are two bytes long.

Command Format:



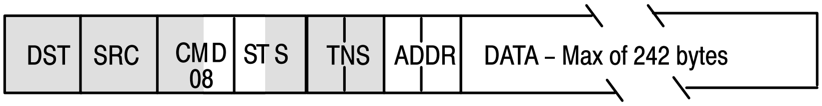
Reply Format:



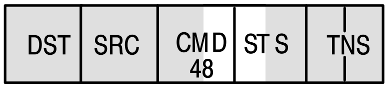
Unprotected Write

This command writes words of data into any area of PC data table memory.

Command Format:



Reply Format:



PLC COMMANDS

PLC stations can execute all of the commands in the basic command set. They can also execute the following commands, which apply only to PLC controllers:

- Disable outputs
- Enable program
- Enable scan
- Physical read
- Physical write

The above commands are privileged commands. This means that only a computer can initiate them. Their primary purpose is for uploading and downloading PLC memory.

Disable Outputs

This command turns off all outputs of the PLC controller. Use this command to disable the PLC's output before doing a physical write operation.

Command Format:

DST	SRC	CMD	STS	TNS	FNC
		07			00

Reply Format:

DST	SRC	CMD	STS	TNS
		47		

Enable Program

This command returns control of outputs to the PLC ladder diagram program. Use this command to cancel the effect of the disable outputs command.

Command Format:

DST	SRC	CMD	STS	TNS	FNC
		07			01

Reply Format:

DST	SRC	CMD	STS	TNS
		47		

Enable Scan

This command restarts the PLC processor’s program scanner after a physical write operation has been performed. Note that you must always use the enable scan command to restart the PLC processor after a physical write command.

Command Format:

DST	SRC	CMD	STS	TNS	FNC
		07			03

Reply Format:

DST	SRC	CMD	STS	TNS
		47		

Physical Read

This command reads bytes of data from the PC data table or program memory. Use this command up to upload the contents of PLC memory to your computer.

Use the SIZE field to specify the number of bytes to be read. To specify a number of PC words, SIZE should be an even value because PC words are two bytes long.

Command Format:

DST	SRC	CMD	STS	TNS	ADDR	SIZE
		04				

Reply Format:

DST	SRC	CMD	STS	TNS	DATA – Max of 244 bytes
		44			

Physical Write

This command writes bytes of data into the PC data table or program memory. Use this command to download the contents of a computer file into PLC memory.

Use the SIZE field to specify the number of bytes to be written. To specify a number of PC words, SIZE should be an even value because PC words are two bytes long.

Command Format:

DST	SRC	CMD	STS	TNS	ADDR	DATA – Max of 243 bytes
		08				

Reply Format:

DST	SRC	CMD	STS	TNS
		43		

PLC-2 COMMANDS

PLC-2 Stations can execute all of the commands in the basic command set. They can also execute the following commands, which apply only to PLC-2 Family controllers:

- Enter download mode
- Enter upload mode
- Exit download/upload mode
- Physical read
- Physical write
- Set data table size

The above commands are privileged commands. This means that only a computer can initiate them. Their primary use is for uploading and downloading PLC-2 memory.

Enter Download Mode

This command puts the PLC- processor into the download mode. Use this command on a PLC-2 station before attempting to send any physical write commands to the station.

Command Format:

DST	SRC	CMD	STS	TNS	FNC
		07			04

Reply Format:

DST	SRC	CMD	STS	TNS
		47		

When you send an Enter Download/Upload mode command, the Industrial Terminal Port is disabled until you send an Exit Download/Upload mode command. When the Industrial Terminal Port is disabled, it enters the Mode Select State. In order to leave this state,you will have to manually select a mode at the Industrial Terminal.

Enter Upload Mode

If you are using a 1771-KA, series A, revision F module, or a 1771-KG, series A, use the **Enter Download Mode** command 07, function 04, shown above in section “Enter Download Mode,” before sending physical read commands.

If you are using a 1771-KA2 module, or a 1771-KG, series B, use the Enter Upload command 07, function 06, shown below, before sending physical read commands.

Enter Upload Mode

This command puts the PLC-2 processor into the upload mode. Use this command on a PLC-2 station before attempting to send any physical read commands to the station.

Command Format:

DST	SRC	CMD	STS	TNS	FNC
		07			06

Reply Format:

DST	SRC	CMD	STS	TNS
		47		

When you send an Enter download/upload mode command, the industrial terminal port is disabled until you send an Exit Download/Upload mode command. When the industrial terminal port is disabled, it enters the Mode Select State. In order to leave this state, you will have to manually select a mode at the industrial terminal.

Exit Download/Upload Mode

This command takes the PLC-2 processor out of the upload or download mode. Use this command to restart the PLC-2 processor after performing an upload or download operation. If you don't send this command after a download/upload mode command, you will have to recycle power at the 1771-KA or 1771-KA2 module to enable industrial terminal communication.

Command Format:

DST	SRC	CMD	STS	TNS	FNC
		07			05

Reply Format:

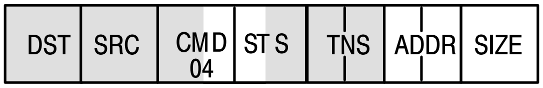
DST	SRC	CMD	STS	TNS
		47		

Physical Read

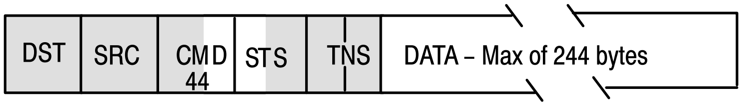
This command reads bytes of data from the PC data table or program memory. Use this command to upload the contents of PLC-2 memory to your computer.

Use the SIZE field to specify the number of bytes to be read. To specify a number of PC words, SIZE should be an even value because PC words are two bytes long.

Command Format:



Reply Format:

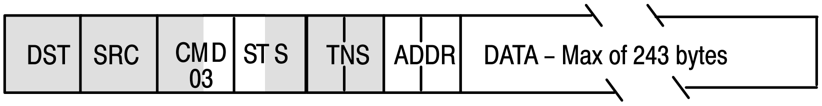


Physical Write

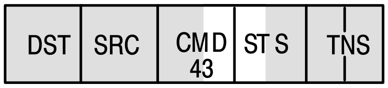
This command writes bytes of data into the PC data table or program memory. Use this command to download the contents of a computer file into PLC-2 memory.

Use the SIZE field to specify the number of bytes to be written. To specify a number of PC words, SIZE should be an even value because PC words are two bytes long.

Command Format:



Reply Format:



Set Data Table Size

This command sets the data table size for the PLC-2 processor. Use this command immediately before performing any physical writes on the PLC-2 processor.

For the DATA field in this command, enter the number of bytes of memory that you want to allocate to the PLC-2 data table. Since PC words are two bytes long, the DATA value is double the number of words in the PLC-2 data table. The DATA value is also equivalent to the physical address (chapter 6) of the start of the processor’s program memory. To determine allowable data table sizes, refer to the programming manual for the appropriate PLC-2 processor.

Command Format:

DST	SRC	CMD	STS	TNS	FNC	DATA
		06			08	

Reply Format:

DST	SRC	CMD	STS	TNS
		46		

PLC-3 Commands

PLC-3 stations can execute all of the commands in the basic command set. They can also execute the following commands, which apply only to PLC-3 controllers:

Non-privileged Commands

- Bit write
- File read
- File write
- Word range read
- Word range write

Privileged Commands

- Download request
- Physical read
- Physical write
- Restart request
- Shutdown request
- Upload request

Only a computer can initiate privileged commands. Their primary use is for uploading and downloading PLC-3 memory.

Only a computer or another PLC-3 station can initiate the non-privileged PLC-3 commands listed above. Their primary use is for transferring data between two PLC-3 files. Those files may be located in the same PLC-3 processor or in two different PLC-3's.

In addition to the message packet fields already described for the basic command set (section titled "Message Packet"), PLC-3 messages may also contain the following fields:

1. ASCII SYMBOL - contains the ASCII codes for the characters in a PLC-3 symbolic address. Chapter 6 gives more information on the format of symbolic addresses.

2. EXT STS - contains extended status information in a reply message. This field is used only if the STS value if F0 hex.

3. PACKET OFFSET - contains the word offset between the DATA field of the current message packet and the DATA field of the first packet in the transaction. This offset value appears only in command messages for file transfers, where the complete transaction might require more than one command or reply message packet. The value of PACKET OFFSET is zero for the first packet in a transaction.

4. PLC-3 LOGICAL ADDRESS - contains the logical address of a PLC-3 file or word. Chapter 6 gives more details about the contents of the address field.

5. PLC-3 PHYSICAL ADDRESS - contains the logical address of a PLC-3 file or word. Chapter 6 gives more details about the contents of this address field.

6. TOTAL TRANS - contains the total number of data words transferred by the current transaction. This is the total number of PLC-3 words to be transferred in the DATA fields of all message packets in the current transaction. TOTAL TRANS appears only in command messages for file transfers, where the transaction might take more than one message packet to complete.

7. W/F - is the word/file symbolic address flag. This one-byte flag specifies whether the symbolic address field following it represents a word address. The value of W/F is zero if the symbol represents a word address and non-zero if the symbol represents a file address.

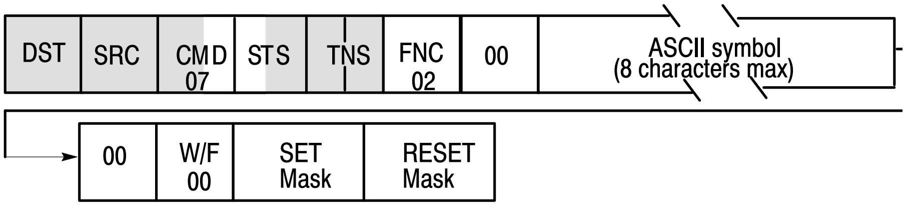
8. WORD OFFSET - contains the word offset between the desired word and the beginning of the addressed file. The offset is zero for the first word of a file. In word-range read and write commands, this field can be combined with a symbolic file address to specify a word address.

Bit Writes

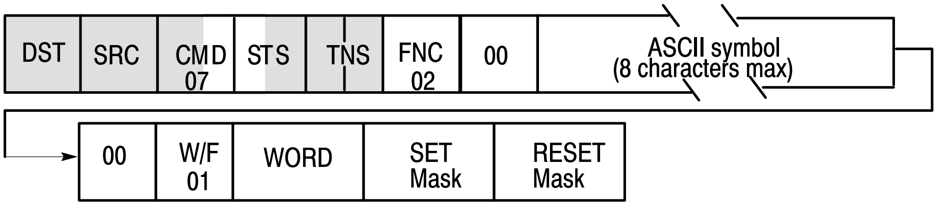
This is a bit write command to modify the bits at the address specified by either a word symbol, a file symbol plus a word offset, or a logical address. This address must point to a word within a file. The function code is 2. Unlike the current unprotected and protected bit writes in the basic command set, this command can be change the bits in a single word only.

Command Format:

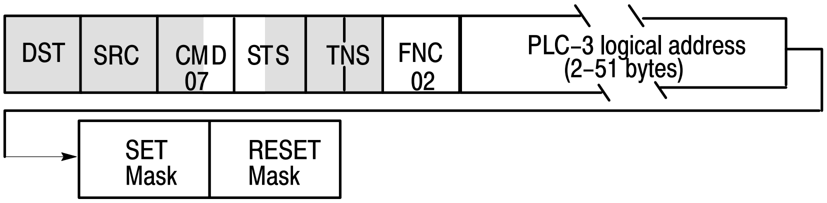
A. Word symbol address



B. File symbol address plus word offset



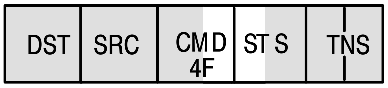
C. Logical address



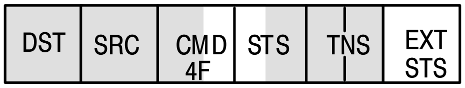
Reply Format:

This is the same as the reply packet format for all current unprotected, protected, and privileged bit writes.

A. Format when successful execution



B. Format when reporting an error

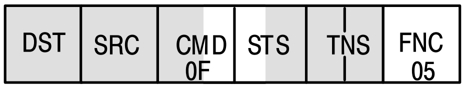


Where the extended status byte is optional.

Download Request

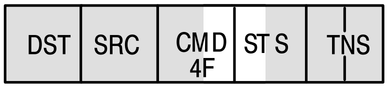
A computer can use this command to inform the 1775-KA module that it wants to do a download. If the 1775-KA module grants the download privilege, the computer may begin issuing physical reads or writes. If a different station already has the download privilege, the second station is denied the privilege. The function code is 5.

Command Format:

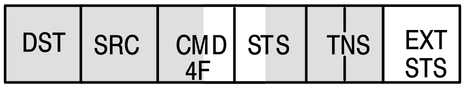


Reply Format:

A. Format when successful execution



B. Format when reporting an error



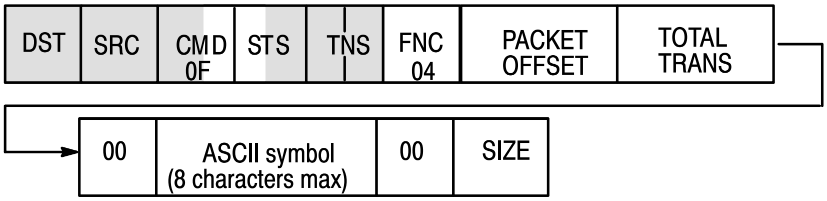
Where the extended status byte is optional.

File Read

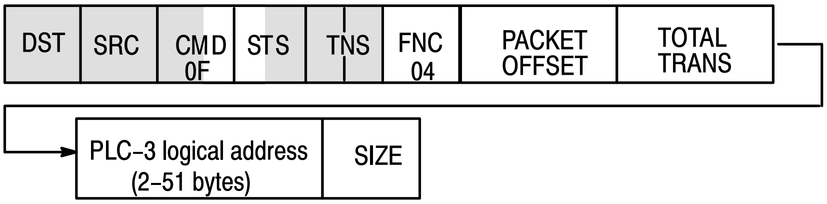
This is a read command whose starting address is either a file symbol or a block address. This starting address must point to a file of words. The function code is 4.

Command Format:

A. File symbol address



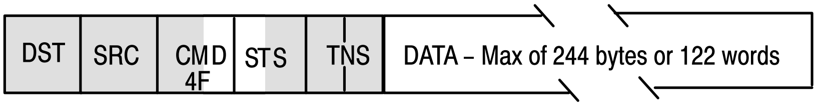
B. Logical address



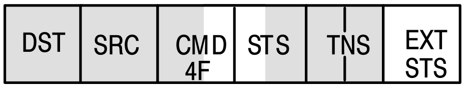
Reply Format:

This is the same as the reply packet format for all current unprotected, protected, and privileged reads.

A. Format when successful execution



B. Format when reporting an error



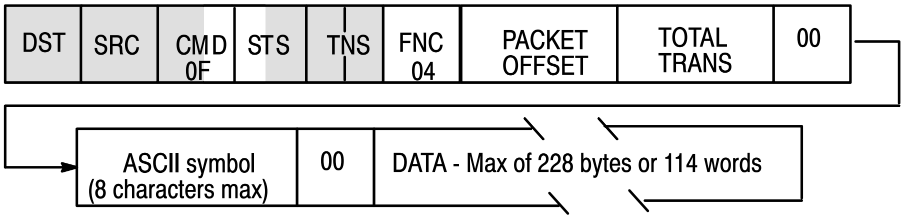
Where the extended status byte is optional.

File Write

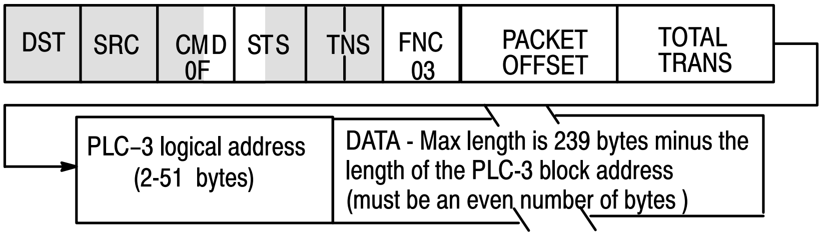
This is a write command whose starting address is either a file symbol or a block address. This starting address must point to a file of words. The function code is 3.

Command Packet Format:

A. File symbol address



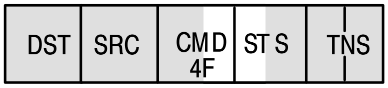
B. Logical address


DST
SRC
CMD
0F
STS
TNS
FNC
03
PACKET
OFFSET
TOTAL
TRANS
PLC–3 logical address
(2-51 bytes)
DATA - Max length is 239 bytes minus the length of the PLC-3 block address (must be an even number of bytes)

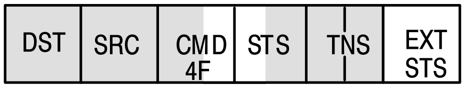
Reply Format:

This is the same as the reply packet format for all current unprotected, protected, and privileged writes.

A. Format when successful execution



B. Format when reporting an error

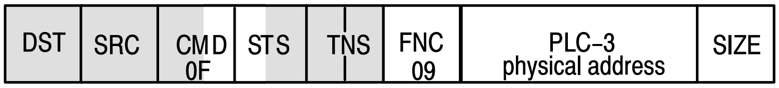


Where the extended status byte is optional.

Physical Read

This is a read command where the starting address is a PLC-3 physical address. It is used to upload from a PLC-3 to a computer. The destination 1775-KA module will accept this command only after the source station has successfully transmitted a shutdown request. The function code for this command is 9.

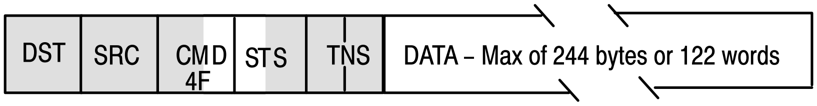
Command Format:



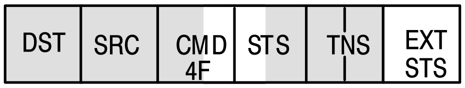
Reply Format:

This is the same as the reply packet format for all current unprotected, protected, and privileged reads.

A. Format when successful execution



B. Format when reporting an error

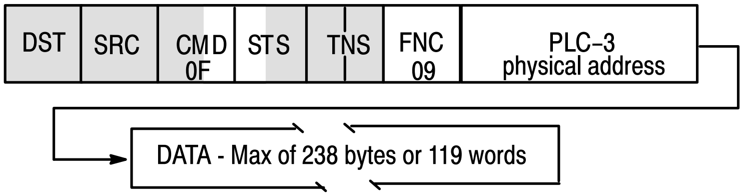


Where the extended status byte is optional.

Physical Write

This is a write command where the starting address is a PLC-3 physical address. It is used to download to a PLC-3 from a computer. The destination 1775-KA module will accept this command only after the source station has successfully transmitted a shutdown request. The function code for this command is 8.

Command Format:



Reply Format:

This is the same as the reply packet format for all current unprotected, protected, and privileged writes.

A. Format when successful execution

DST	SRC	CMD 4F	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4F	STS	TNS	EXT STS

Where the extended status byte is optional.

Restart Request

This command is used by the computer to terminate an upload or a download. The computer cannot issue this command until after it has successfully completed an upload or download operation with the destination station. This command causes the 1775-KA module to revoke the upload and download privileges for the source computer station and to initialize a PLC-3 restart. The function code for this command is 10 decimal.

Command Format:

DST	SRC	CMD 0F	STS	TNS	FNC 0A

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4F	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4F	STS	TNS	EXT STS

Where the extended status byte is optional.

Shutdown Request

This command is used by the computer to ask the 1775-KA module to initiate either a PLC-3 shutdown (if the computer has download privileges) or a freeze on file allocations (if the computer has upload privileges). The computer cannot issue this command until it has successfully transmitted an upload or download request to the 1775-KA module. This command has a function code of 7.

Command Format:

DST	SRC	CMD 0F	STS	TNS	FNC 07

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4F	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4F	STS	TNS	EXT STS

Where the extended status byte is optional.

Upload Request

This command is used by the computer to inform the 1775-KA module that it wants to do an upload. If the module grants the upload privilege, the computer may begin issuing physical reads. If a different station already has the upload privilege, the second station is denied the privilege. The function code is 6.

Command Format:

DST	SRC	CMD 0F	STS	TNS	FNC 06

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4F	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4F	STS	TNS	EXT STS

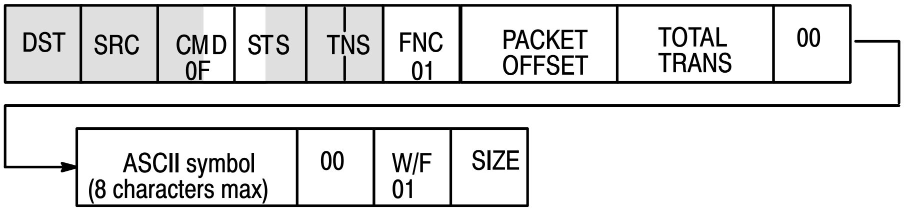
Where the extended status byte is optional.

Word Range Read

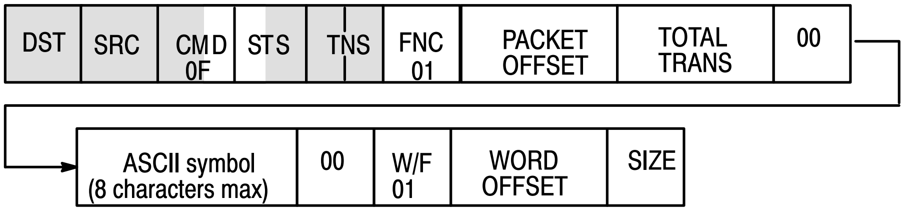
This is a read command whose starting address is either a word symbol, a file symbol plus a word offset, or a block address. This starting address must point to a word in a file. The function code is 1. A special case of this command is the single-word read, where the number of bytes to read is only two bytes (one word).

Command Format:

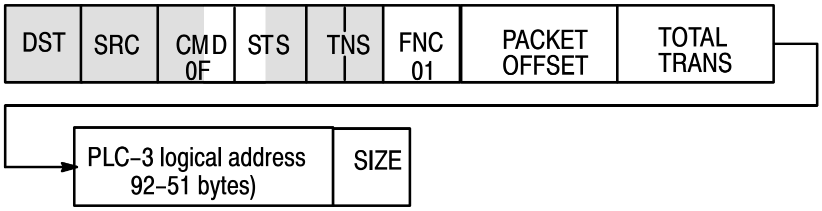
A. Word symbol address



B. File symbol address plus word offset



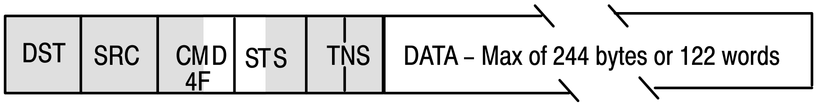
C. Logical address



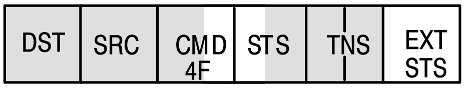
Reply Format:

This is the same as the reply packet format for all current unprotected, protected, and privilege reads.

A. Format when successful execution



B. Format when reporting an error



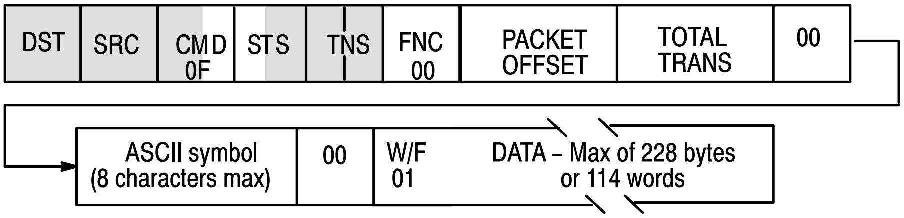
Where the extended status byte is optional.

Word Range Write

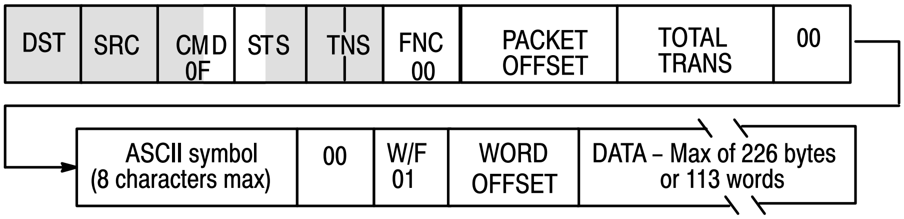
This is a write command whose starting address is either a word symbol, a file symbol plus a word offset, or a logical address. This starting address must point to a word in a file. The function code is 0 (zero). A special case of this command is the single word write, where the data field is only one word long.

Command Format:

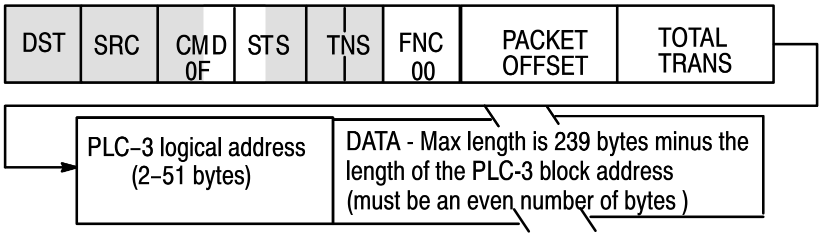
A. Word symbol address


DST
SRC
CMD 0F
STS
TNS
FNC 00
PACKET OFFSET
TOTAL TRANS
00
ASCII symbol (8 characters max)
00
W/F 01
DATA – Max of 228 bytes or 114 words

B. File symbol address plus word offset



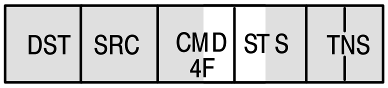
C. Logical address


DST
SRC
CMD 0F
STS
TNS
FNC 00
PACKET OFFSET
TOTAL TRANS
PLC–3 logical address (2–51 bytes)
DATA - Max length is 239 bytes minus the length of the PLC-3 block address (must be an even number of bytes)

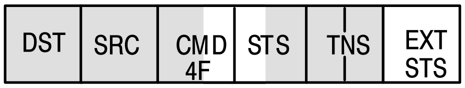
Reply Format:

This is the same as the reply pack format for all current unprotected, protected, and privileged writes.

A. Format when successful execution



B. Format when reporting an error



Where the extended status byte is optional.

PLC-4 Commands

PLC-4 stations can execute all of the commands in the basic command set. They can also execute the following commands, which apply only to PLC-4 Microtrols:

- Allocate processor
- De-allocate processor
- Initialize processor
- Physical read
- Physical write
- Physical write with mask
- Set to program mode
- Set to run mode
- Set to single scan test mode
- Set to test mode

The above are privileged commands. This means that only a computer can execute them. Their primary purpose is for uploading and downloading PLC-4 memory.

Note that PLC-4 stations cannot initiate any type of command message.

In addition to the message packet fields already described for the basic command set (section titled “Message Packet”), PLC-4 messages may also contain a byte called PLC-4 SEL. This field specifies which controller in the PLC-4 Microtrol loop is the ultimate destination of the command message. The allowed values for this field are:

PLC-4 SEL value	Meaning
0	Selects controller #1
1	Selects controller #2
2	Selects controller #3
3	Selects controller #4
4	Selects controller #5
5	Selects controller #6
6	Selects controller #7
7	Selects controller #8

Allocate

This command causes the 1773-KA module to allocate PLC-4 access privileges to the computer station that originated the allocate commands. Once the computer has this access privilege, it can send any of the other privileged commands to the selected PLC-4 controller. No other computer can gain access privileges to the same PLC-4 controller until the privileges of the first computer have been de-allocated.

If the 1773-KA module loses power, all access privileges are de-allocated.

Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 05	PLC-4 sel

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS

B. Format when reporting n error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

De-allocate

This command de-allocates access privileges to the selected PLC-4 controller.

Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 06	PLC-4 sel

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

Initialize Processor

This command initializes, or clears, memory in the selected PLC-4 controller. Use this command to clear the data table memory of the controller before downloading to it. The controller must be in program load mode and must not be executing any other privileged command.

Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 0C	PLC-4 sel

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

Physical Read

This command reads the specified number of bytes from the memory of the selected PLC-4 controller. Use this command to upload the contents of PLC-4 memory to your computer. The SIZE field contains the number of bytes to be read, and it may have a value from 1 to 242 decimal.

Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 0D	PLC–4 sel	ADDR	SIZE

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS	DATA – 1–242 bytes

B. Format when reporting an error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

Physical Write

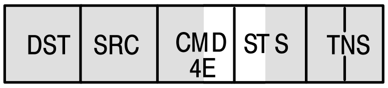
This command writes bytes of data into the memory of the selected PLC-4 controller. Use this command to download data from your computer to the controller. The PLC-4 ADDRESS field contains the physical byte address where the data will begin being written into PLC-4 memory. Refer to chapter 6 for a description of this physical address.

Command Format:

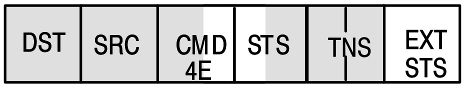
DST	SRC	CMD 0E	STS	TNS	FNC 0E	PLC–4 sel	PLC–4 address	DATA - 1-239 bytes

Reply Format:

A. Format when successful execution



B. Format when reporting an error



Where the extended status byte is optional.

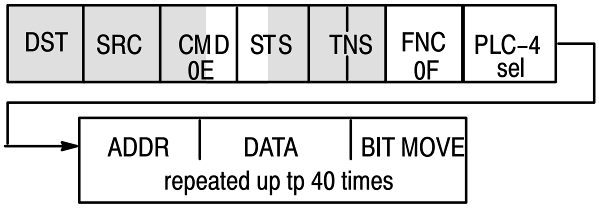
Physical Write With Mask

This command sets or resets bits in a memory word of the selected controller. The ADDRESS field contains the physical PLC-4 address of the memory word to be modified. Refer to chapter 6 for a description of physical addresses.

The BIT MASK field specifies which bits in the PLC-4 word are to be modified, and the DATA field specifies whether those bits are to be set or reset. For each bit that is 1 in the BIT MASK, the corresponding bit is the addresses PLC- 4 word is set to the same value (1 or 0) as the corresponding bit in the DATA field. For each bit that is 0 in the BIT MASK, the corresponding bit of the addresses PLC-4 word is left unchanged.

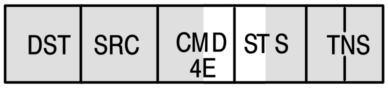
Note that you can modify up to 40 PLC-4 words in this way by specifying a series of ADDRESS, DATA, and BIT MASK fields.

Command Format:

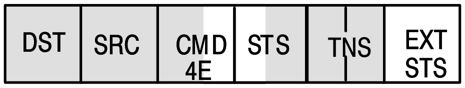


Reply Format:

A. Format when successful execution



B. Format when reporting an error



Where the extended status byte is optional.

Set To Program Mode

This command sets the selected controller to the Program Load mode.
Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 01	PLC–4 sel

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

Set To Run Mode

This command sets the selected controller to Run mode.

Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 02	PLC–4 sel

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

Set To Single Scan Test Mode

This command sets the selected controller to Single Step Test mode.

Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 04	PLC–4 sel

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

Set To Test Mode

This command sets the select controller to Test mode.

Command Format:

DST	SRC	CMD 0E	STS	TNS	FNC 03	PLC–4 sel

Reply Format:

A. Format when successful execution

DST	SRC	CMD 4E	STS	TNS

B. Format when reporting an error

DST	SRC	CMD 4E	STS	TNS	EXT STS

Where the extended status byte is optional.

Chapter 6

Data Manipulation

General

This chapter explains two areas of special concern when you are transmitting messages between computers and PC's:

- Data encoding
- Addressing formats

This information contained in this chapter gives some application details that relate to the data and address fields of the message formats described in chapter 5.

Data Encoding

In general, PC store binary data (1s and 0s) in 16-bit groups called words. If you are looking at this data from a computer, however, you may interpret it in a number of different ways, depending on your application needs.

Number Systems

You may use any one of the following number systems to represent data in your computer application programs:

- Binary
- Binary coded decimal
- Decimal
- Hexadecimal
- Octal

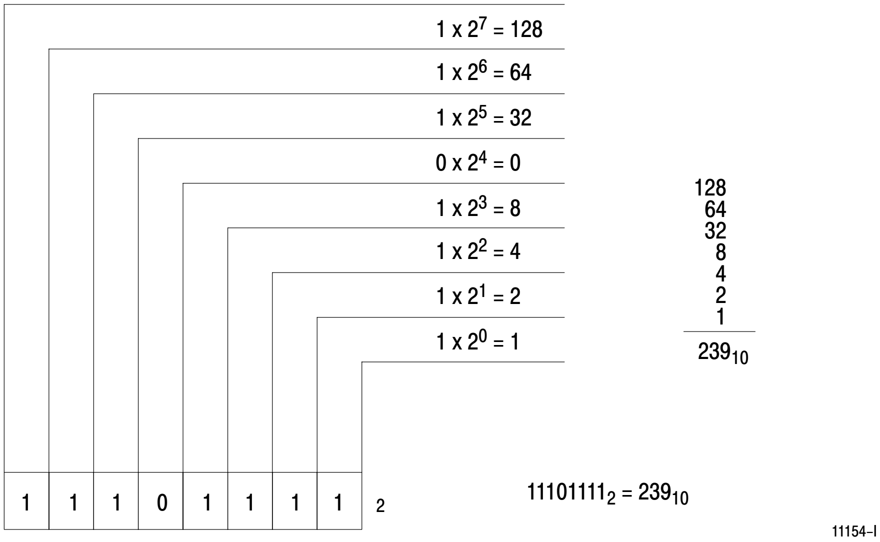
You must design your computer application programs to make any necessary conversions from one number system to another. Once you have selected the number system that is best for your applications, try to use only that one system and convert all data values to that base to avoid confusion.

Binary

The binary number system is probably the simplest to use for computer and PC applications because it is the most natural way to represent data bits. However, since the binary system uses only the digits 0 and 1, it is cumbersome to show values in binary format.

Each digit in a binary number has a certain place value expressed as a power of 2. You can calculate the decimal equivalent of a binary number by multiplying each binary digit by its corresponding place value and then adding the results of the multiplications. Figure 6.1 shows the binary representation of the decimal number 239.

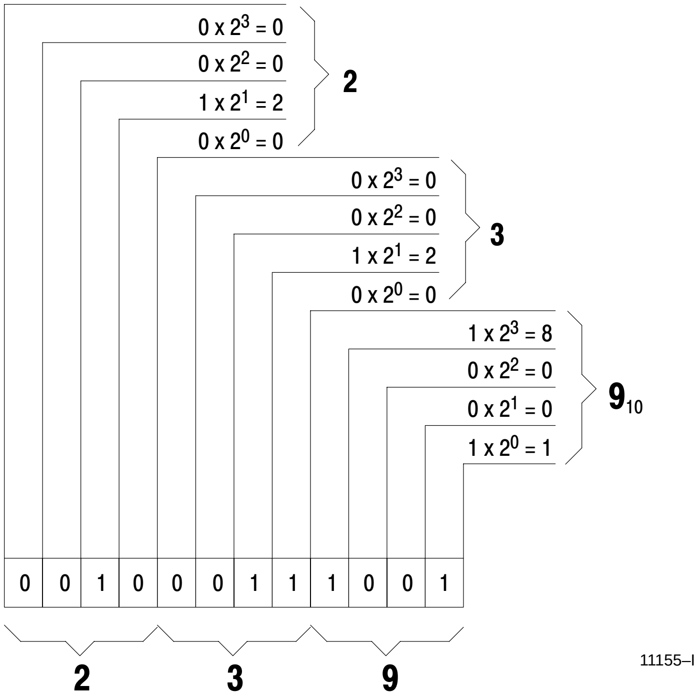
Figure 6.1
Binary Numbers



Binary Coded Decimal

Quite often, PC data is represented in binary coded decimal (BCD) form. In this system, each group of four bits in a PC word represents one decimal number between 0 and 9. In this way, each 16-bit word can represent a BCD value between 0 and 9,999. Figure 6.2 shows the BCD representation of the decimal number 239.

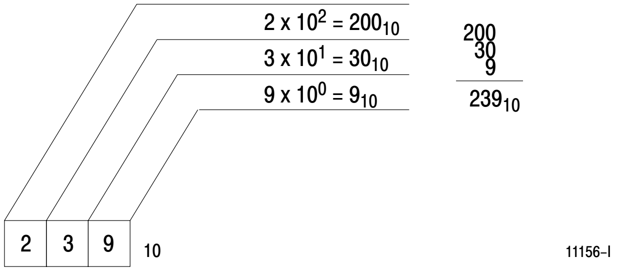
Figure 6.2
BCD Representation of Decimal 239



Decimal

The decimal number system is probably the easiest for us to use because it is most familiar to us. It uses the common digits 0 through 9, and each digit has a place value that is a power of 10 (Figure 6.3). However, despite the convenience of decimal numbers, it is often easier to convert binary data to a number system other than decimal.

Figure 6.3
Decimal Representation, Number 239



Hexadecimal

The hexadecimal number system is the most compact way to represent binary data, and it allows for the easiest conversion to and from binary. This system uses a number set of 16 digits: the numbers 0 through 9 and the letter A through F (where the letter A through F are equivalent to the decimal numbers 10 through 15, respectively).

Each group of four data bits represents one hexadecimal digit between 0 and F. In this way, each 16-bit data word can have a hexadecimal value between 0 and FFFF.

Each digit of a hexadecimal number has a place value that is a multiple of 16. To convert a hexadecimal number to its decimal equivalent, multiply each hexadecimal digit by its corresponding place value and add the results of the multiplications. Figure 6.4 shows the hexadecimal representation of the decimal number 423.

Octal

The octal number system is also a relatively easy way to represent binary data. This system uses the eight digits 0 through 7.

Each group of three data bits represents one octal digit between 0 and 7. This factor presents a slight conversion problem because bytes and words usually contain an even number of bits. Thus, an 8-bit byte can have an octal value between 0 and 377, while 16-bit word can have an octal value between 0 and 177777.

Each digit of an octal number has a place value that is a multiple of 8. To convert from octal to decimal, multiply each octal digit by its corresponding place value and add the results of the multiplications. Figure 6.5 shows the octal representation of the decimal number 239.

Figure 6.4
Hexadecimal Numbers

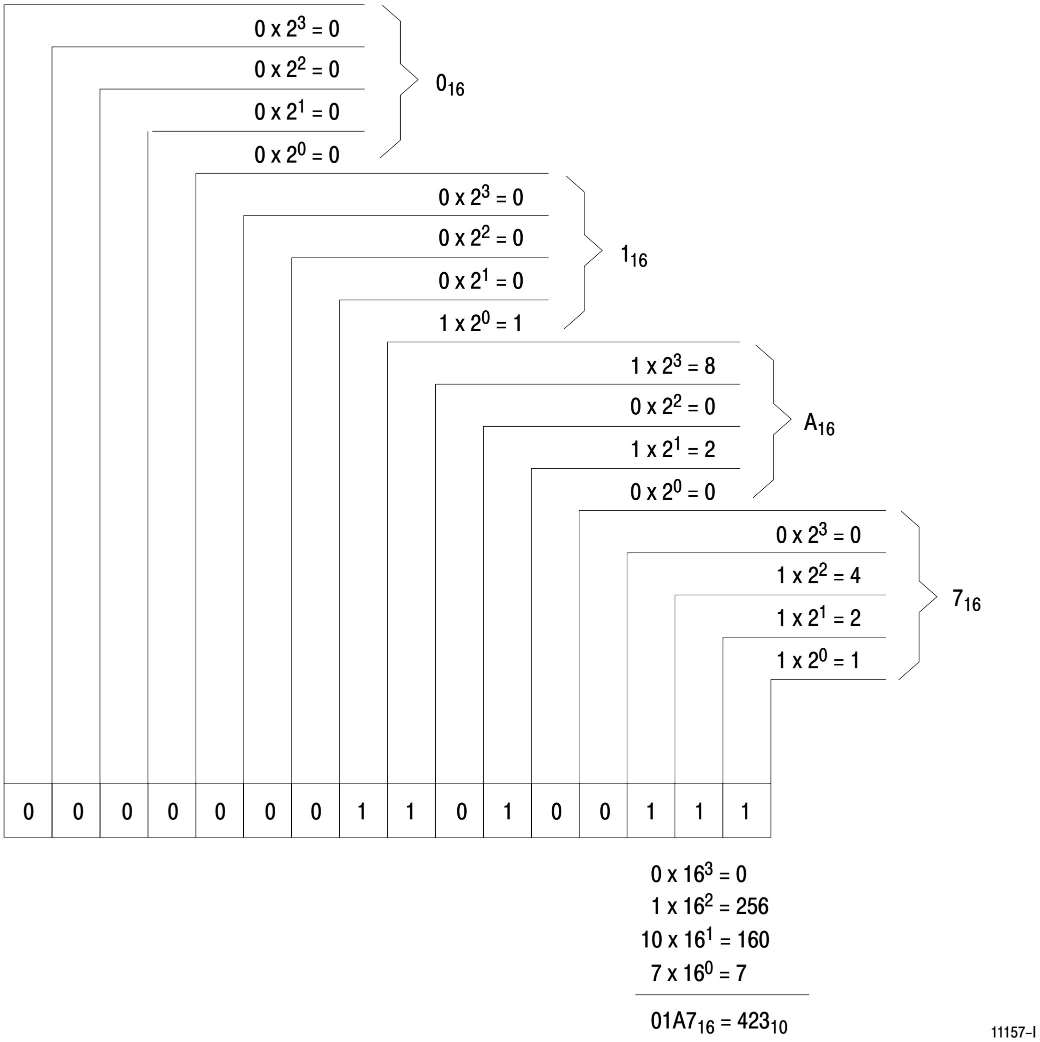
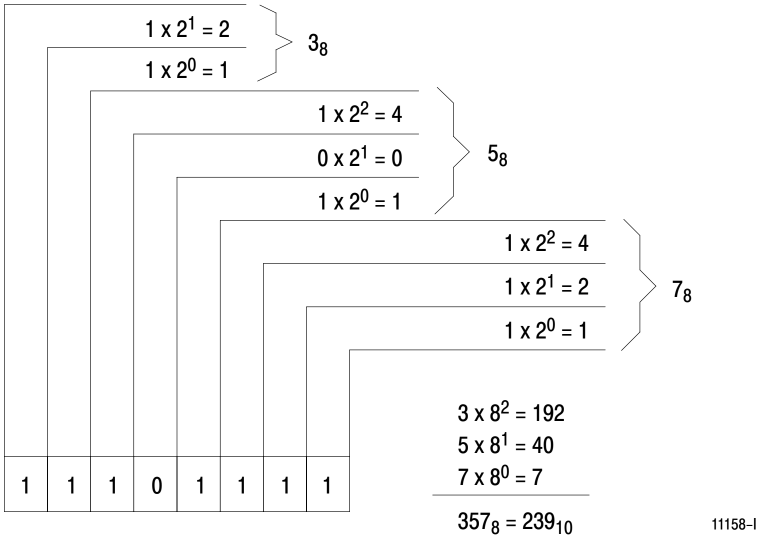


Figure 6.5
Octal Numbers



Order of Transmission

PCs store data in 16-bit (2-byte) words. The bits in these words are numbered (addressed) 0 through 17 octal, going from right to left within a word, as follows:

PC Word

Bits	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00

In PC memory, the words are arranged as shown above. However, when the KE/KF module transmits data over its RS-232-C link, it transmits one byte at a time. The module always transmits the low byte (bits 00 through 07) of a word before the high byte (bits 10 through 17) of the same word. Also, UART transmits the low bit first within a byte. Thus, when a PC word is traveling over the RS-232-C link, it will look like this:

	First Byte									Second Byte							
Bits	07	06	05	04	03	02	01	00		17	16	15	14	13	12	11	10

Time ⟶

This does not present a problem at PC stations on the link because PCs always store and retrieve their data in the same order of low byte first. It can, however, require you to do some extra computer application programming to maintain the proper byte and word order in PC data stored in the computer.

Three factors that can influence the ability of your computer to handle PC data are:

- The size of words in your computer's memory
- The left-to-right or right-to-left ordering of bits within a word in your computer's memory
- Whether the computer considers the low order byte of a word to have an even or an odd address

If your computer uses something other than 2-byte, 16-bit words, you should design your application programs to make the proper conversions from PC word addresses to computer word addresses. When stored in a computer, each PC word should start on an even byte boundary.

Figure 6.6a shows a 16-bit word in PC memory.

Figure 6.6b shows a 16-bit computer word with right-to-left byte and bit order (as in DEC PDP-11/34 or VAX 11/780). It also represents a 16-bit word in an 8-bit processor (such as Zilog Z-80 or Intel 8086 microprocessor). If your computer has this type of word order, the conversion is straightforward.

Figure 6.6c shows a 16-bit computer word with left-to-right byte and bit order (as in IBM Series 1). If your computer has this type of word order, the conversion is more complex. You will have to swap bytes into and out of buffers.

Figure 6.6d shows a 16-bit computer word with left-to-right byte order and right-to-left bit order (as in Zilog Z8000 or Motorola 68000 microprocessors). If your computer has this type of word order, your communication driver must handle the task of byte swapping as it loads data into a buffer. Successive bytes received from the PC must be stored in the following order:

1, 0, 3, 2, 5, 4, 7, 6, 9, 8,...

Addressing

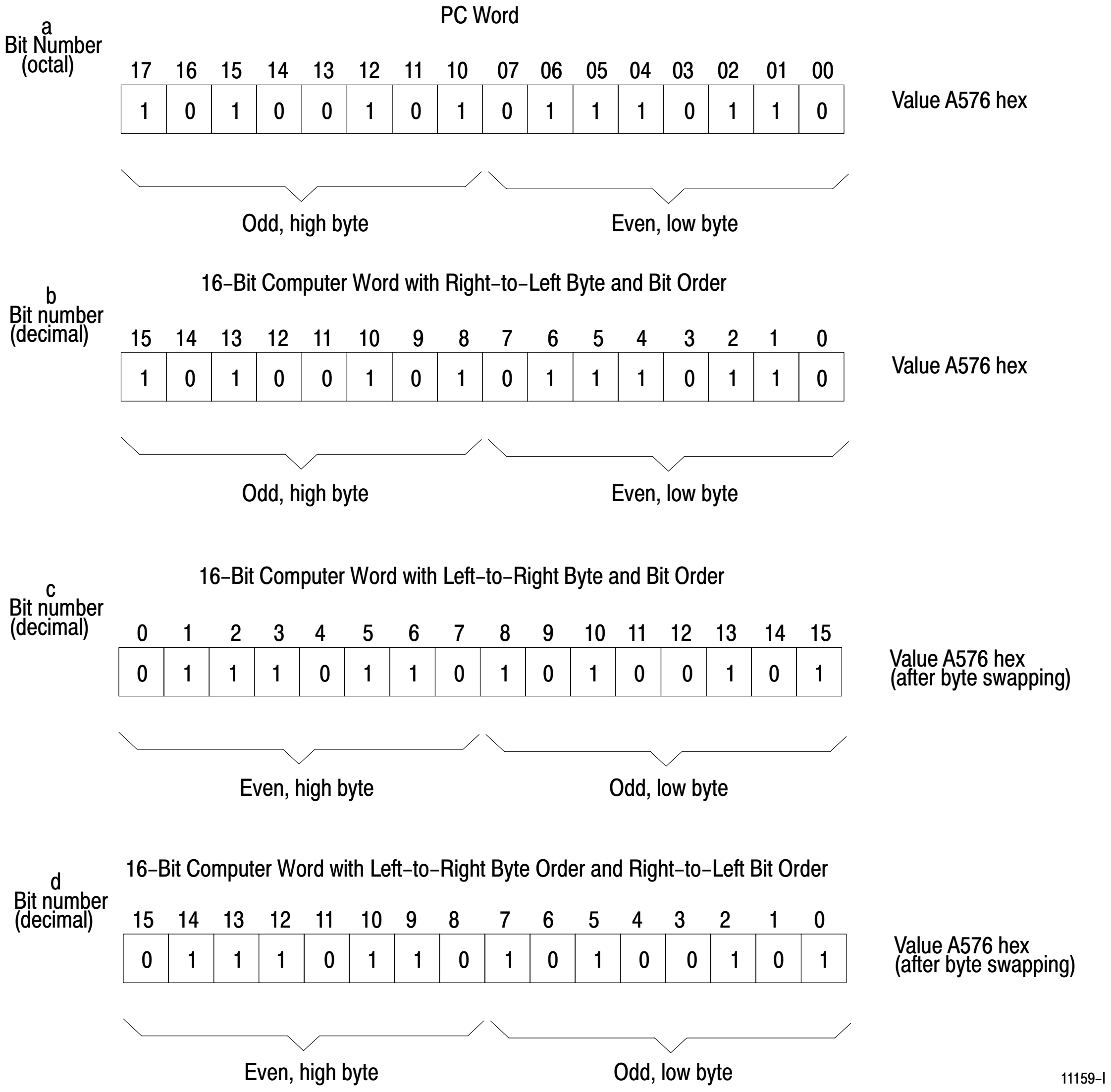
There are three types of addressing a computer can use in command message that it transmits to PC stations:

- Logical
- Physical
- Symbolic

Logical Addressing

Logical addressing refers to the type of addressing that a PC uses in its ladder diagram program to access its own data table memory. This is the same type of addressing you would use in non-privileged commands (that is, in commands that access only PC data table memory). Because of the differences in PC memory organization, the logical addressing scheme varies with controller type.

Figure 6.6
Results of Transmitting low Byte First



PC Word

a Bit Number (octal)	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00	
	1	0	1	0	0	1	0	1	0	1	1	1	0	1	1	0	Value A576 hex

16-Bit Computer Word with Right-to-Left Byte and Bit Order

b Bit number (decimal)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	1	0	1	0	0	1	0	1	0	1	1	1	0	1	1	0	Value A576 hex

16-Bit Computer Word with Left-to-Right Byte and Bit Order

c Bit number (decimal)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
	0	1	1	1	0	1	1	0	1	0	1	0	0	1	0	1	Value A576 hex (after byte swapping)

16-Bit Computer Word with Left-to-Right Byte Order and Right-to-Left Bit Order

d Bit number (decimal)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	0	1	1	1	0	1	1	0	1	0	1	0	0	1	0	1	Value A576 hex (after byte swapping)

PLC/PLC-2

PLC and PLC-2 Family controllers access their data tables by using an octal word address. In PLC/PLC-2 command messages, this type of logical word address must be represented as an equivalent byte address. This byte address appears in the 2-byte field labeled ADDR in the message block formats (chapter 5).

To encode a logical PLC/PLC-2 address, convert the octal word address to whatever number system you are using in your computer application programs. Next, double this converted word address to get the corresponding byte address. Place the result in the ADDR field, low byte first.

For example, to address PLC word 020, you would first convert the octal value 20 to the desired base. In this example, let’s use hexadecimal values. Octal 20 is 10 hex. Doubling this value gives 20 hex for the byte address. You would then code the value 0020 hex in the ADDR field of the message, low byte first. In binary format, ADDR would look like:

ADDR

First Byte

0	0	1	0	0	0	0	0

Low Byte
(value 20 hex)

Second Byte

0	0	0	0	0	0	0	0

High Byte
(value 00 Hex)

NOTE: PLC and PLC-2 family controllers use this same logical addressing format when they transmit command messages to another station. If you plan to transmit a command message to your computer from one of these PCs, you should set up a buffer space in your computer to simulate PC memory. You would then have to write computer application programs to accept and execute commands from the PC stations and to translate the ADDR value into the corresponding address in the simulated PC memory.

PLC-3

PLC-3 controllers use a form of logical addressing known as extended addressing. With extended addressing, you specify the address of each level (or subdivision) of PLC-3 memory, down to the smallest subdivision you want to access. You can use this method to specify up to 6 levels of PLC-3 extended addressing, which is enough to give the address of a particular word in PLC-3 memory.

To send a command message to a PLC-3 station, you would put the extended address in the field labeled “PLC-3 extended addr” in the message block formats (chapter 5). Figure 6.7 shows an example of how to enter a PLC-3 extended address in this message field.

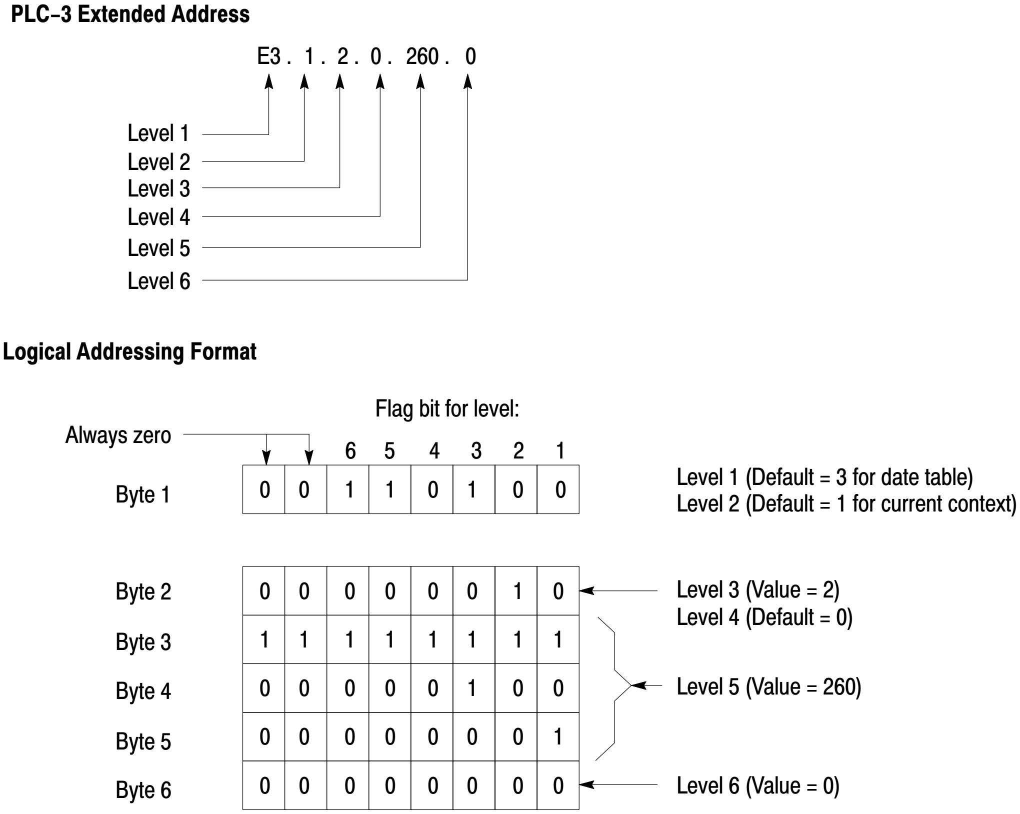
The first byte in the extended address field is a set of bit flags that indicate which addressing levels are specified in the bytes following the bit flags. If a flag bit is set to 1, the address bytes must contain a specification for the corresponding level of the extended address. If a flag bit is zero, the address bytes should not contain a specification for that addressing level; instead, a default value is assumed.

For Level:	The default address is:
1	3 (data table)
2	Current context
All others	0

You must always specify the value for the last level of the desired extended address, even if it is the default value.

If the address values can be specified in one byte each, then you can code the values directly. If it takes two bytes to specify an address, then you must use a delimiter byte of value FF hex before each 2-byte address. Any 2-byte value should be encoded low byte first.

Figure 6.7
Example of PLC-3 Logical Addressing Format



			6	5	4	3	2	1
Byte 1	0	0	1	1	0	1	0	0
Byte 2	0	0	0	0	0	0	1	0
Byte 3	1	1	1	1	1	1	1	1
Byte 4	0	0	0	0	0	1	0	0
Byte 5	0	0	0	0	0	0	0	1
Byte 6	0	0	0	0	0	0	0	0

Byte 1 –– is the flag byte. In this case it indicates that the addresses for levels 3, 5, and 6 are specified in the bytes that follow. Default values are used for the levels 1, 2, and 4.

Byte 2 –– is the value of the level–3 address.

Byte 3 –– is a delimeter that says the next two bytes are one address.

Byte 4 –– is the low byte of the level–5 address.

Byte 5 –– is the high byte of the level–5 address. Note that bytes 4 and 5 together give a value of 260 for the level–5 address.

Byte 6 –– is the value of the level–6 address. Even though it is the default value, it must be specified because it is the last level in the desired extended address.

11160–I

In Figure 6.7, the first byte contains the bit flags to indicate which addressing levels are specified. In this example, only levels 3, 5, and 6 are specified; default values are used for the other levels.

In Figure 6.7, the level-5 address is 260 (decimal), which is too large to fit in one byte. Therefore, a byte of all 1's is used to delimit the 2-byte address value for this level. The value 260 is then coded low byte first. Note that the last level (level 6 in this example) must be specified in the address field even though it is equal to the default value of zero.

NOTE: PLC-3 controllers can also accept PLC/PLC-2 type command messages with the PLC/PLC-2 logical addressing format. Before sending this type of command to a PLC-3 station, you must first allocate a PLC-3 input file to simulate PLC/PLC-2 memory (refer to publication 1775-802). In addition, PLC-3 controllers can transmit both PLC-3 and PLC/PLC-2 type command messages, each with its appropriate logical addressing format. Therefore, if you plan to transmit commands from a PLC-3 to your computer, you should set up a computer buffer to simulate a PLC-3 file and write computer application programs that are capable of interpreting all the types of addressing formats that will appear in the command messages.

PLC-4 Microtrol

PLC-4 Microtrol controllers use a form of logical addressing that specifies the identification number of the controller in the loop, the section of memory, and the bit address. To specify a PLC-4 logical address in a command message, you would use the appropriate binary code listed in Table 6.A. Note that these binary codes let you address only PLC-4 words, not bits.

PLC-4 controllers can accept only PLC/PLC-2 type non-privileged commands. The binary code for the PLC-4 logical address goes in the 2-byte field labeled ADDR in the PLC/PLC-2 message block formats (chapter 5). Put the low byte (bits 0 through 7) of the binary address code into the first byte of ADDR.

For example, suppose we want to address storage word 4 in controller 3 in a PLC-4 Microtrol loop. The binary code for this address is:

00000000 00110100

In the command message that accesses this storage word, the ADDR field would be:

ADDR

First Byte

0	0	1	0	0	0	0	0

Second Byte

0	0	0	0	0	0	0	0

Table 6.A
Binary Codes for PLC-4 Logical Addresses

PLC-4 Data Table Location	word ref'd	Binary Code															
		Bit															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Unused *	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X
Storage Word #1	1-7	0	0	0	0	0	0	0	0	0	0	0	1	P	P	P	H
Storage Word #2	1-6	0	0	0	0	0	0	0	0	0	0	1	0	P	P	P	H
Storage Word #3	1-5	0	0	0	0	0	0	0	0	0	0	1	1	P	P	P	H
Storage Word #4	1-4	0	0	0	0	0	0	0	0	0	1	0	0	P	P	P	H
Storage Word #5	1-3	0	0	0	0	0	0	0	0	0	1	0	1	P	P	P	H
Storage Word #6	1-2	0	0	0	0	0	0	0	0	0	1	1	0	P	P	P	H
Storage Word #7	1	0	0	0	0	0	0	0	0	0	1	1	1	P	P	P	H
Unused *	0	0	0	0	0	0	0	0	0	1	0	0	0	X	X	X	X
I/O, Flags, Word #1	1-4	0	0	0	0	0	0	0	0	1	0	0	1	P	P	P	H
I/O, Flags, Word #2	1-3	0	0	0	0	0	0	0	0	1	0	1	0	P	P	P	H
I/O, Flags, Word #3	1-2	0	0	0	0	0	0	0	0	1	0	1	1	P	P	P	H
I/O, Flags, Word #4	1	0	0	0	0	0	0	0	1	1	1	0	0	P	P	P	H
I/O, Flags Area	32	0	0	0	0	0	0	0	0	1	1	0	1	P	P	P	0
Timer/Ctr 1: Acc	1-64	0	0	0	0	0	0	0	0	1	1	1	0	P	P	P	H
Pre	1-63	0	0	0	0	0	0	0	0	1	1	1	1	P	P	P	H
Timer/Ctr 2: Acc	1-62	0	0	0	0	0	0	0	1	0	0	0	0	P	P	P	H
Pre	1-61	0	0	0	0	0	0	0	1	0	0	0	1	P	P	P	H
Timer/Ctr 3: Acc	1-60	0	0	0	0	0	0	0	1	0	0	1	0	P	P	P	H
Pre	1-59	0	0	0	0	0	0	0	1	0	0	1	1	P	P	P	H
Timer/Ctr 4: Acc	1-58	0	0	0	0	0	0	0	1	0	1	0	0	P	P	P	H
Pre	1-57	0	0	0	0	0	0	0	1	0	1	0	1	P	P	P	H
Timer/Ctr 5: Acc	1-56	0	0	0	0	0	0	0	1	0	1	1	0	P	P	P	H
Pre	1-55	0	0	0	0	0	0	0	1	0	1	1	1	P	P	P	H
Timer/Ctr 6: Acc	1-54	0	0	0	0	0	0	0	1	1	0	0	0	P	P	P	H
Pre	1-53	0	0	0	0	0	0	0	1	1	0	0	1	P	P	P	H
Timer/Ctr 7: Acc	1-52	0	0	0	0	0	0	0	1	1	0	1	0	P	P	P	H
Pre	1-51	0	0	0	0	0	0	0	1	1	0	1	1	P	P	P	H
Timer/Ctr 8: Acc	1-50	0	0	0	0	0	0	0	1	1	1	0	0	P	P	P	H
Pre	1-49	0	0	0	0	0	0	0	1	1	1	0	1	P	P	P	H
Timer/Ctr 9: Acc	1-48	0	0	0	0	0	0	0	1	1	1	1	0	P	P	P	H
Pre	1-47	0	0	0	0	0	0	0	0	1	1	1	1	P	P	P	H
Timer/Ctr 10: Acc	1-46	0	0	0	0	0	0	1	0	0	0	0	0	P	P	P	H
Pre	1-45	0	0	0	0	0	0	1	0	0	0	0	1	P	P	P	H
Timer/Ctr 11: Acc	1-44	0	0	0	0	0	0	1	0	0	0	1	0	P	P	P	H
Pre	1-43	0	0	0	0	0	0	1	0	0	0	1	1	P	P	P	H
Timer.Ctr 12: Acc	1-42	0	0	0	0	0	0	1	0	0	1	0	0	P	P	P	H
Pre	1-41	0	0	0	0	0	0	1	0	0	1	0	1	P	P	P	H
Timer/Ctr 13: Acc	1-40	0	0	0	0	0	0	1	0	0	1	1	0	P	P	P	H
Pre	1-39	0	0	0	0	0	0	1	0	0	1	1	1	P	P	P	H
Timer/Ctr 14: Acc	1-38	0	0	0	0	0	0	1	0	1	0	0	0	P	P	P	H
Pre	1-37	0	0	0	0	0	0	1	0	1	0	0	1	P	P	P	H
Timer/Ctr 15: Acc	1-36	0	0	0	0	0	0	1	0	1	0	1	0	P	P	P	H
Pre	1-35	0	0	0	0	0	0	1	0	1	0	1	1	P	P	P	H

PLC-4 Data Table Location		word ref'd	Binary Code															
										Bit								
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Timer/Ctr 16:	Acc	1-34	0	0	0	0	0	0	1	0	1	1	0	0	P	P	P	H
	Pre	1-33	0	0	0	0	0	0	1	0	1	1	0	1	P	P	P	H
Timer/Ctr 17:	Acc	1-32	0	0	0	0	0	0	1	0	1	1	1	0	P	P	P	H
	Pre	1-31	0	0	0	0	0	0	1	0	1	1	1	1	P	P	P	H
Timer/Ctr 18:	Acc	1-30	0	0	0	0	0	0	1	1	0	0	0	0	P	P	P	H
	Pre	1-29	0	0	0	0	0	0	1	1	0	0	0	1	P	P	P	H
Timer/Ctr 19:	Acc	1-28	0	0	0	0	0	0	1	1	0	0	1	0	P	P	P	H
	Pre	1-27	0	0	0	0	0	0	1	1	0	0	1	1	P	P	P	H
Timer/Ctr 20:	Acc	1-26	0	0	0	0	0	0	1	1	0	1	0	0	P	P	P	H
	Pre	1-25	0	0	0	0	0	0	1	1	0	1	0	1	P	P	P	H
Timer/Ctr 21:	Acc	1-24	0	0	0	0	0	0	1	1	0	1	1	0	P	P	P	H
	Pre	1-23	0	0	0	0	0	0	1	1	0	1	1	1	P	P	P	H
Timer/Ctr 22:	Acc	1-22	0	0	0	0	0	0	1	1	1	0	0	0	P	P	P	H
	Pre	1-21	0	0	0	0	0	0	1	1	1	0	0	1	P	P	P	H
Timer/Ctr 23:	Acc	1-20	0	0	0	0	0	0	1	1	1	0	1	0	P	P	P	H
	Pre	1-19	0	0	0	0	0	0	1	1	1	0	1	1	P	P	P	H
Timer/Ctr 24:	Acc	1-18	0	0	0	0	0	0	1	1	1	1	0	0	P	P	P	H
	Pre	1-17	0	0	0	0	0	0	1	1	1	1	0	1	P	P	P	H
Timer/Ctr 25:	Acc	1-16	0	0	0	0	0	0	1	1	1	1	1	0	P	P	P	H
	Pre	1-15	0	0	0	0	0	0	1	1	1	1	1	1	P	P	P	H
Timer/Ctr 26:	Acc	1-14	0	0	0	0	0	1	0	0	0	0	0	0	P	P	P	H
	Pre	1-13	0	0	0	0	0	1	0	0	0	0	0	1	P	P	P	H
Timer/Ctr 27:	Acc	1-12	0	0	0	0	0	1	0	0	0	0	1	0	P	P	P	H
	Pre	1-11	0	0	0	0	0	1	0	0	0	0	1	1	P	P	P	H
Timer/Ctr 28:	Acc	1-10	0	0	0	0	0	1	0	0	0	1	0	0	P	P	P	H
	Pre	1-9	0	0	0	0	0	1	0	0	0	1	0	1	P	P	P	H
Timer/Ctr 29:	Acc	1-8	0	0	0	0	0	1	0	0	0	1	1	0	P	P	P	H
	Pre	1-7	0	0	0	0	0	1	0	0	0	1	1	1	P	P	P	H
Timer/Ctr 30:	Acc	1-6	0	0	0	0	0	1	0	0	1	0	0	0	P	P	P	H
	Pre	1-5	0	0	0	0	0	1	0	0	1	0	0	1	P	P	P	H
Timer/Ctr 31:	Acc	1-4	0	0	0	0	0	1	0	0	1	0	1	0	P	P	P	H
	Pre	1-3	0	0	0	0	0	1	0	0	1	0	1	1	P	P	P	H
Timer/Ctr 32:	Acc	1-2	0	0	0	0	0	1	0	0	1	1	0	0	P	P	P	H
	Pre	1	0	0	0	0	0	1	0	0	1	1	0	1	P	P	P	H

Legend:

“words ref'd” These are the number of words which can be read or written to a given location.

* These codes are unused because of “holes” in the address space of PLC-2 family products (0 to 7 and 100 to 107 octal.) Reads or writes to these codes will be rejected in error.

** These addresses are illegal for Bit Write commands.

X X X “DON'T CARE” bits.

P P P This is a bit pattern that selects the desired controller in the PLC-4 Microtrol loop; “000” selects controller #1, 001” selects controller #2, etc.

H Selects high byte when 1, low byte when zero.

Physical Addressing

Physical addressing is the type of addressing a computer would use to send a privileged command to a PC station. In particular, you would use physical addressing to upload or download PC memory. The recommended procedure for doing this is to use a series of physical read or write commands that begin at physical address 0000 and proceed sequentially to the end of PC memory.

Because of the differences in PC memory organization, the physical addressing scheme varies somewhat with controller type.

PLC

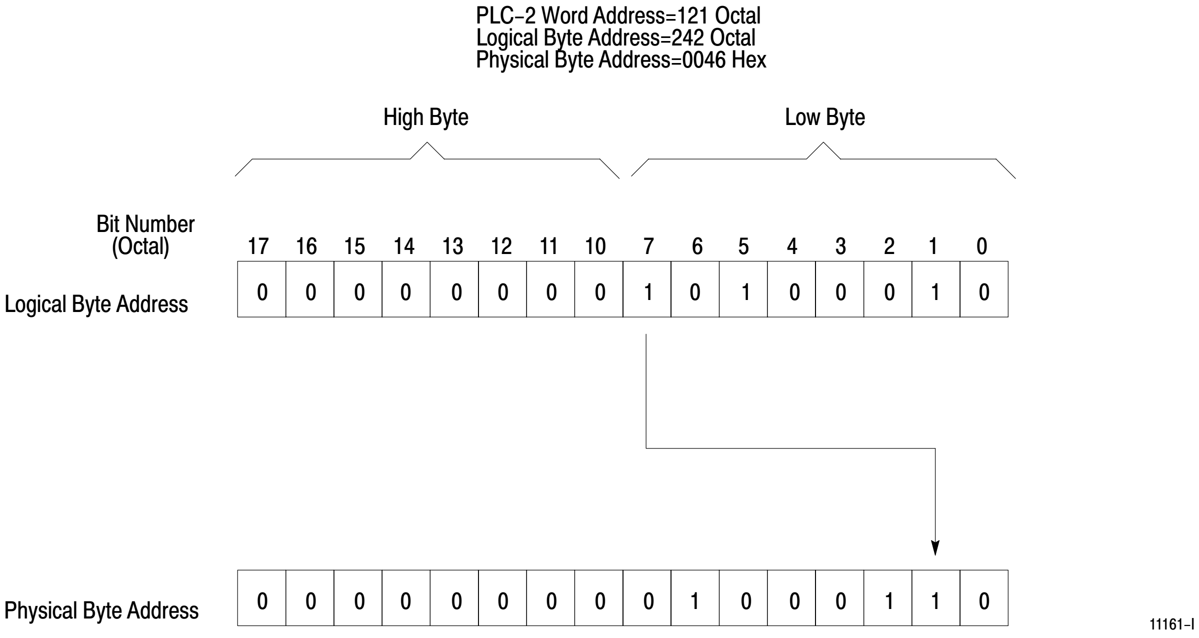
PLC controllers use physical addresses that are exactly the same values as the corresponding logical addresses. Remember that the logical address is a byte address, so the physical address will also be a byte address. For example, the logical byte address of the 17th word in PLC memory is 32 decimal, and the physical address of the word is also 32 decimal.

To send a physical read or write command to a PLC station, put the PLC physical address in the ADDR field of the command message format (chapter 5). Be sure to encode the low byte of the physical address as the first byte in the ADDR field.

PLC-2

PLC-2 controllers use physical addresses that are directly related to the logical addresses. To convert a given logical address to its corresponding physical address, move bit 7 of the logical address to bit position 1 and shift bits 1 through 6 to the left one position. Figure 6.8 illustrates the conversion process for logical word address 121. Remember that the logical PLC-2 address is a byte address, so the physical address will also be a byte address.

Figure 6.8
Converting PLC-2 Logical to Physical Address



To send a physical read or write command to a PLC-2 station, put the PLC-2 physical address in the ADDR field of the command message format (chapter 5). Be sure to encode the low byte of the physical address as the first byte in the ADDR field.

PLC-3

PLC-3 controllers use physical addresses that are related to logical addresses by means of pointers. Since no two PLC-3 systems are configured identically, the pointers are not fixed. Therefore, there is no algorithm for converting logical to physical PLC-3 addresses.

The PLC-3 physical address is a word address. It goes in the 4-byte field labeled “PLC-3 physical addr” in the PLC-3 physical read or write command message format (chapter 5). The format for this physical address field is:

PLC–3 Physical Address

First byte	A24	A23	A22	A21	A20	A19	A18	A17
Second byte	0	0	0	0	0	0	0	0
Third byte	A8	A7	A6	A5	A4	A3	A2	A1
Fourth byte	A16	A15	A14	A13	A12	A11	A10	A9

In this format, A1 through A24 represent the 1 to 24 bits of the physical address value. For example, to address a command message to physical word address 12,200 decimal (002FA8 hex), you would use the following binary code in the address field:

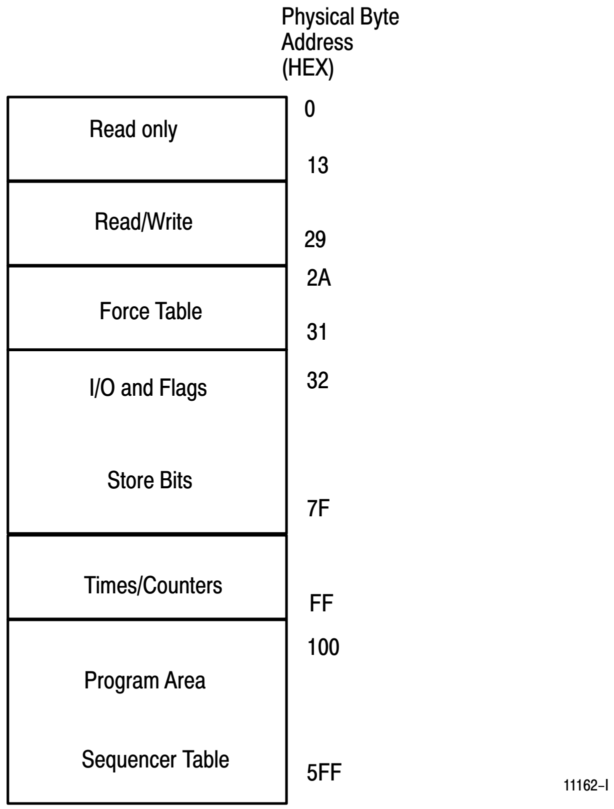
First byte	0	0	0	0	0	0	0	0	(value 00 hex)
Second byte	0	0	0	0	0	0	0	0	(always 00 hex)
Third byte	1	0	1	0	1	0	0	0	(value A8 hex)
Fourth byte	0	0	1	0	1	1	1	1	(value 2F hex)

The recommended procedure for uploading or downloading PLC-3 memory is to begin at physical address 0000 and proceed sequentially to the end of memory. Since a single physical read or write command can transfer only about 120 words of data, it will take many such commands to upload or download the entire PLC-3 memory. Therefore, each successive physical read or write should begin at the next physical address after the one where the previous command stopped.

PLC-4 Microtrol

PLC-4 Microtrol controllers use physical addresses that begin at 00 for the first word of memory and continue to 5FF hex for the last word of memory. Figure 6.9 is a map of PLC-4 physical memory.

Figure 6.9
PLC-4 Physical Memory



Specifying a physical address in PLC-4 privileged commands requires 3 bytes. The first byte is the identification number for a particular controller in the loop, and it is labeled “PLC-4 sel” in the command message formats (chapter 5). The next two bytes contain the physical address of a word in the selected controller’s memory, and they are labeled ADDR in the command message formats (chapter 5). When encoding the physical address, be sure to put the low byte of the address value in the first byte of the ADDR field.

Symbolic Addressing

Symbolic addressing uses ASCII symbols to represent a logical address. Only PLC-3 controllers can accept symbolic addresses. Before using a symbolic address in a message, you must first define the symbol at the PLC-3 that is to receive the message (refer to publication 1775-801).

The symbolic address field can be from 1 to 8 bytes long. The first byte contains the ASCII code for the first character in the symbol name, the second byte contains the ASCII code for the second character, and so on. If the symbol name is more than 8 characters long, encode only the first 8 characters.

To use a symbolic address in a command message, encode the symbol in the field labeled "ASCII symbol" in the command message formats (chapter 5). Note that the message format show a byte of value zero before and after the symbolic address field. You must include these zero bytes because they act as delimiters to distinguish the symbolic address from other fields in the message.

NOTE: PLC-3 controllers can also transmit commands that contain symbolic addresses. If you plan to transmit this type of command message to your computer from a PLC-3 station, then you must write computer application programs that are capable of accepting these commands and interpreting the symbolic addresses.

Chapter 7

Error Reporting

General

This chapter contains a list of error codes associated with Data Highway start-up and run-time situations. The error information is available in three places: the PC application program, the internal error counters in each module, and the STS byte in the header of a message packet. The first section deals with the PC application program's error word and is the most accessible to the user.

The second section lists internal error counters, bytes of information stored in RAM in each Data Highway module. These numbers can be read only by issuing a diagnostic read command from a device connected to one of the modules that supports an RS-232-C port.

The section titled "Transmissions Between Computer and Full-Duplex Modules" explains the STS byte that is contained in the command header. This data is presented as part of the RS-232 protocol and is available only to modules that support this (e.g., KE/KF module).

A brief note: Error codes pointing to traffic problems or other ambiguous situations should be dealt with only after a troubleshooter has thoroughly tested the highway cabling for shorts and bad connections. Most application problems can be traced initially to bad cabling. What can be very frustrating is that bad cabling does not necessarily mean that a point is completely shorted or completely open. Some traffic may get through (narrowed bandwidth), but the frequency of re-tries on a given message becomes very high.

ERROR WORD in User Programming (1771-KG, 1771-KA, and 1774-KA Modules)

An error code word specified in the header rung is the primary source of explanation for programming problems and runtime problems. Error codes are stored in this word for most events that can be observed by a Data Highway user.

It is important to note that the display of an error code at a given location does not necessarily mean a faulted condition exists on the highway. Due to the nature of the polling algorithm and the built-in re-try and recovery procedures, random noise or contention can easily be ignored if normal procedures are followed. For example, Error 89 is primarily an indication of heavy use at one node, rather than a faulted condition. The appearance of a code like Error 89 is then a user application question, and should be dealt with by understanding the trade-offs between performance and node utilization. Obviously pre-scan errors (1-29) indicate a program problem and should be fixed immediately; but merely trapping and halting on run-time errors, without understanding the relationship of the error to loading factors and node utilization, does not allow efficient operation of the Data Highway.

A troubleshooter should also make full use of the counter (high-byte in the error word) to record frequency of errors. This contributes to any application solution that requires redistribution of node traffic.

The error code word is four BCD digits wide. The uppermost digit is not displayed by the Industrial Terminal in the GET instruction in the header rung, but this is the least valuable part of the error code and can often be ignored. If necessary, this number can be easily displayed using GET BYTE and a PUT. The lower two digits are always the error number. Each number indicates a different condition, which is explained in the paragraphs below. The meaning of the upper two digits varies depending on the error number.

There are about 50 different error numbers that can be broken up into two major and four minor groups:

> Pre-scan Errors: These are mostly communication zone syntax errors. They are numbered 1 to 29. They are detected as the KA module pre-scans the communication zone just before it starts scanning start bits.
>
> Note that no syntax errors are detected in the header rung. If the pre-scanner does not recognize the header rung it will decide that it is not a header rung, and will continue scanning until the end of memory or until it finds a valid header rung.
>
> Pre-scan errors always cause the module to turn on the PROG light and enter an error state. When the keylock is turned to PROGRAM the error state is exited and the PROG light turned off.
>
> For all pre-scan errors the upper two digits act as a pointer to the rung containing the error. Rungs are numbered from 1 to 99, then wrap around back to 0 through 99, and so on. The first rung after the header rung is number 1.
>
> If no pre-scan error is detected, the error code word is set to all zeros at the end of the pre-scan.
>
> Runtime Errors: For all runtime errors the first two digits of the code are a modulo 100 error counter. The first error is number 1.
>
> Except as noted, the module continues operating after recovering from a runtime error.

Runtime errors are divided into the following groups:

1. Message formatting: numbered from 30 to 39. They are detected as a message is being formatted and before it is sent. A few of these errors result in the KA turning on the PROG light and entering the error state.
2. Reply errors: numbered between 50 and 59, and detected by the local station when a reply is received.
3. Remote errors: numbered from 80 to 89, and returned in the reply message from the remote station as the result of errors in execution of a message at the remote station. It should be noted that a Data Highway module will continue to generate replies as long as it is functioning properly. These errors are contained in the STS byte of the reply message.
4. Local errors: numbered 90 to 99 and detected by the local station; they are the result of being unable to properly transmit on the highway.

Error numbers are listed and explained below:

- 01: No longer used. In revisions previous to F for 1771-KA this code could be set as a result of intermittent hard errors on the KA to PC cable.

- 02: The pre-scan aborted because the test codes in RACK0 are bad. This error should never occur, because if the test codes are ever bad the PC should fault before the KA ever gets to the pre-scan.

- 03: The KA or KG does not have enough internal memory to store the start bit index. This should only occur if the communication rung is very large. To correct this, the communication zone can be shortened, perhaps by combining commands or windows. It may also occur if a RAM hardware error corrupts the internal memory organization.

- 04: The first element of a memory access branch is not a GET. This condition can only be detected if no command rungs have yet been detected, and if the branch being processed started with a BST. (BST = Branch Start; BND = Branch End)

- 05: The address field of the first GET in a memory access branch contains an invalid Data Highway station address. The address must be less than 377.

- 06: The second element of a memory access branch is not a GET.

- 07: The third element of a memory access branch is not a GET.

- 08: Invalid window address in memory access branch: the address in the second GET is greater than the third GET.

- 09: There is something besides a BST or BND after the third GET of a memory access branch.

- 10: There is something other than a BST or output instruction after a BND in a memory access rung.

- 11: A rung in the communication zone starts with something besides a BST or an XIC, or the zone contains no command rungs and the terminating rung has been omitted.

- 12: The start bit address specified for a command rung is in the low byte of a word. Start bits must always be in the 10 to 17 range.

- 13: The second element of a command rung is not an XIC.

- 14: The command code (the low digit of the address) of the second XIC in a command rung is invalid.

- 15: The station address in the second XIC of a command rung is invalid. Station address must be less than 377.

- 16: An instruction or an opcode other than XIC, XIO, BST, or BND has been encountered in a bit control command rung, or the output element of the rung is incorrect.

- 17: The third element of a read or write command rung is not a GET.

- 18: The fourth element of a read or write command rung is not a GET.

- 19: The fifth element of a read or write command rung is not a GET.

- 20: The sixth element of a read or write command rung is not a legal output instruction.

- 21: The start of a command rung is not an XIC, or the terminating rung has been omitted. This error is only generated if valid command rungs precede it.

- 22: The word containing start/done/error bits is positioned so that it crosses a boundary in the data table. (For example, start bit 07710 means that the error bits would be in word 100, or 17710 means that the error bits would be located in the first word of the program).

- 23: An unprotected write or bit control command is specified and the DIP switch that enables sending of unprotected commands is off.

- 24: Not used in 1771-KA.

- 25: Not used in 1771-KA.

- 26: Too many command rungs. A maximum of 255 command rungs can be programmed.

- 27: The default timeout is too large. Valid timeouts are from 011 to 407 octal. The value 010 causes the timeout to be disabled.

- 30: A command rung syntax error has been detected in RUN mode after the pre-scan. This means that either the communication zone has been altered by online editing, hardware problems, or a highway download operation, or that it has been moved by a gap or ungap of a rung before the zone, or by changing the data table size.

- 31: This will not be generated by rev F (1771-KA). Previous modules generated this code if the start bit scanner detected a hard error on the PC-to-module cable.

- 32: The data block of a read or write command is too large to fit in one Data Highway message. This error causes the module to enter an error state.

- 33: An invalid command code was detected by the start bit scanner. This has the same cause as error #30. This error causes the module to enter an error state.

- 34: An invalid station address was detected by the start bit scanner. See error #30. This error causes the module to enter an error state.

- 35: The KA attempted to send an unprotected write or bit control command while the DIP switch that enables unprotected commands was off. This can only occur at run time if the DIP switch that controls this option was changed without removing power from the module.

- 36: The start bit was turned off after a message was sent but before the done bit, local error, or remote error bit was set. This is the situation that causes the local error to turn on, then off for 85 ms after the start bit is reset. The error code word is set before the local error bit turns on.

- 37: The start bit was timed out by the automatic module timer before a reply message arrived. This happens for one of the following reasons:

  - Noise on the highway causes loss of a message. In this case the message probably will succeed if it is re-tried.

  - The remote station powered down or was disconnected from the highway while it was processing the message. If the message is re-tried it should get a 92 error.

  - The timeout is too short. The minimum recommended timeout value is 2 seconds. With a resolution of one second this allows the actual timeout to occur as soon as one second or as late as 2 seconds.

  - A highway loading peak caused the timeout to be exceeded. If this is a rare occurrence it might be acceptable to just re-try the message. Small (<100%) increases in the timeout should eliminate this problem.

  - Malfunction of a highway station is causing it to retain mastership for long periods of time. Check for a station with a faulty receiver, **or a bad highway cable**. The internal diagnostic counters will be most helpful in tracking this one.

- 50: Not used. Prior to rev F (1771-KA) this may have occurred if a PC hard error occurred while processing a reply message.

- 51: The reply message contains an invalid rung number. This should never occur unless PCs are sending messages to a computer program that is not yet functioning properly. This error will not cause the setting of an error bit or done bit.

- 52: A reply has been received at a PC that cannot send messages because it has no command rungs. This should never occur unless faulty reply messages are sent by a computer under debug. This error will not cause the setting of an error bit or done bit.

- 53: A reply message has been received for which the start bit is off. This could mean that the user program turned off the start bit before the done bit came on, or that the automatic timeout is too short. This error often follows a 36 or 37 error. This error will not cause the setting of an error bit or done bit.

- 54: Runtime command rung syntax check failed. This is probably the result of on-line editing. This error will cause the module to enter the error state. See error 30.

- 55: Reply received while PROG light is on or during program mode. This will possibly occur if messages are being executed remotely at the time a runtime syntax check fault occurs, or if the keylock is turned to program mode while messages are being executed. This will not cause the setting of an error bit or done bit.

- 56: The sequence number in a reply message does not match the one being waited for. This is to be expected if the start bit is turned on, a message is sent, then the start bit is turned off and on again before the done bit is set. This error will not cause the setting of an error bit or done bit.

- 57: A reply message has an incorrect size. This should not occur except during the debug of a new highway computer program. This will not cause the setting of a done bit or error bit.

- 81: This error is sent from the remote station if the command message was incorrect. This includes the command code, subcommand code, and size of the command or the requested reply size. This error results in the setting of the remote error bit for the associated rung. This error code corresponds with STS code 10.

- 82: Not used for 1771-KA. The meaning of this code has been assigned to verification errors within the host PC. This error code corresponds with STS code 20.

- 83: Some condition exists at the remote PC that requires manual intervention. This error code corresponds with STS code 30.

- ▪ The cable between the module and the PC is unplugged.

- ▪ The PC is faulted.

Either results in setting the remote error bit for the associated rung.

- 84: Execution of a message at the remote station was aborted because of a hard communication error on the cable or on backplane access between the module and the PC. This error results in the setting of the remote error bit for the associated rung. This error code corresponds with STS code 40.

- 85: An attempt to access an illegal address in the remote PC has aborted message execution. Illegal accesses may result from:

- ▪ Access outside the data table as defined at the remote station.

- ▪ Access outside a memory access window (protected commands only).

Either results in setting the remote error bit for the associated rung. This error code corresponds with STS code 50.

- 86: Execution of a command is disabled at the remote station by a DIP switch option. This error results in setting the remote error bit for the associated rung. This error code corresponds with STS code 60.

- 87: The remote PC is in PROGRAM or REMOTE PROGRAM mode, or the remote KA is in download mode. This error results in setting the remote error bit for the associated rung. This error code corresponds with STS code 70.

- 88: Execution of protected commands at the remote station is inhibited because its PROG light is on. This error results in setting the remote error bit for the associated rung. This error code corresponds with STS code 80.

- 89: The remote station has no memory to store messages. This error will only be signalled after 5 re-tries at half second intervals. it indicates that either a very heavy traffic load is being presented to the remote station, or that the dynamic memory of the remote station is corrupted. If the problem clears up after cycling power and does not recur, the cause may be RAM or CPU failures triggered by heat or noise. If the problem recurs repeatedly the probable cause is too many messages. This error code corresponds with STS code 90.

- 91: The RS-232-C port is not connected to a device or the DSR pin is not being asserted (high = true). This error code corresponds with STS code 01.

- 92: The local station cannot confirm delivery of a command message to the remote station. This does not necessarily mean that the message was not sent or that the done bit or remote error bit will not be set. It is possible, but not probable, that the message will be executed at the remote station. The local error bit will be set by this error. If the done or remote error bits are set they will occur after the local error bit. Reasons for this error include:

- Disconnection of the Data Highway
- Noise on the highway
- Bad cabling or connections
- Remote station is powered off
- Remote station does not exist
- Wrong station address is being used
- Remote module is faulted
- Some module on the highway is in a “bus hog” condition, and prevents proper operation of the highway.

If an error 92 persists, the internal diagnostic counters should be used to pinpoint the location of the faulty module or cable. This error code corresponds with STS code 02.

- 93: This error will occur if the local module, while attempting to send a message, detects another master on the highway. The local error bit for the rung that sent the message will be set. Like error 92, this error does not mean that the message was not received. It is quite probable that the message will be executed. If it is, the done bit or remote error bit may be set after the local error bit if the start bit stays on. If the start bit is unlatched by the local error bit, a 53 or 56 error may result. Occurrence of this error is probably related to general highway conditions, not to the message or the stations sending and receiving it. The message should be re-tried. Possible causes of this error are:

  - Connection of two operating highways
  - Connection of a powered up module to an operating highway
  - Noise on the highway

- Bad cabling or connections

Error 93 is rare but not impossible on a well functioning highway. If it persists, the internal diagnostic counters should be used to pinpoint the location of the faulty module or cable. This error code corresponds with STS code 03.

Local and Remote Error Bits

As a diagnostic tool, these bits are useful to identify the rung that caused an 80- or 90-series error.

The remote error bit indicates that a message was received from the remote station that some condition there prevented successful execution of the message. When a remote error is indicated there is probably nothing wrong with the local station, or with the highway cable. The remote station is most likely working properly. The first causes to investigate are indicated by the various codes.

Setting of a local error bit indicates that the local module is not able to confirm delivery of the command message to the remote station, or that the command rung timed out before the reply message arrived. A local error does not necessarily mean that the message was not received at its destination or that it will not be executed.

Error Codes for 1775-KA

This section describes error codes that the 1775-KA module will report to the PLC-3 application program. Errors are of three types:

- Local
- Reply
- Remote

Local errors are those that the 1775-KA module encounters while trying to execute one of its own message procedures.

Reply errors are those that the 1775-KA module inserts in the STS byte of a reply that it generates in response to a command from a remote station.

Remote errors are those that are returned to the 1775-KA module in a reply message from a remote Data Highway station.

NOTE: The frame of reference in this chapter is that of the 1775-KA All error codes listed here are a result of some action of a 1775-KA.

Error 93 is rare but not impossible on a well functioning highway. If it persists, the internal diagnostic counters should be used to pinpoint the location of the faulty module or cable. This error code corresponds with STS code 03.

Local Error Codes

The 1775-KA module stores local error codes under the user symbol ERROR. Possible local errors are as listed below.

.

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Meaning
		32	The size of the local file involved in a file assignment command is greater than 65,535 bytes.
		34	A station number greater than 376 (octal) was specified for the remote address in an assignment command.
		35	Attempt to send unprotected command is invalid.
		37	The per-packet timeout, which can be set through LIST, ran out before a reply was received. This means that the remote station acknowledged (ACK) the command message, but did not send the reply in the allotted time. (cf. error 92)
	02	92	The remote station specified does not acknowledge (ACK) the message. See discussion of Error 92 in section titled "ERROR WORD in User Programming."
	04	94	Local port is disabled through LIST.
		112	1. Undefined assignment operator in assignment statement. 2. Undefined operator in an expression.
		114	Illegal expression syntax.
		115	Illegal unary (prefix) operator in an expression.
		117	Undefined data following a valid address in a CREATE command, or undefined data following a valid symbol in a DELETE command.
		121	Symbol undefined. This will occur if a symbol appears as the source in an assignment command before it is defined as a symbol. For example, a statement of the form A = A + 6 will give this error if user symbol 'A' has not appeared previously.
		123	System symbol must be a symbolic address. This error will occur if a procedure name is used in place of a symbolic address in an assignment statement or if the system symbol referenced in an assignment doesn't exist.
		124	Illegal destination in an assignment command. This does not necessarily mean that an assignment command was desired because any command line that doesn't look like anything else is assumed to be an assignment command. Lines that will generate this error include: 5 = 4 + 1 6ASDFGHJ Whereas the line WERTYUI will generate an error 140 (unrecognized command).

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Meaning
		125	Illegal modifier for the CREATE command. That is, the command was CREATE/... and the ... was other than LOCAL, GLOBAL, or a legal abbreviation of one of these.
		126	The CREATE command was specified, but the symbol did not begin with an ’@’.
		127	“$” missing in CREATE system symbol address.
		129	Attempt to delete nonexistent symbol.
		140	Unrecognized or ambiguous command (cf. error 124)
		142	Illegal data following GOTO command.
		143	Illegal use of label (e.g., not in a procedure).
		144	Label not found.
		145	Duplicate label. User symbols must be distinct from labels.
		146	Too many nested procedures.
		147	Insufficient privilege for the specified operation. This error can occur when an attempt is made, via the assignment command, to write into a major section of memory in which the 1775-KA module does not have access privileges (namely, major section 0, 1, or 2)
		148	Unbalanced parenthesis in expression.
		149	A procedure name was used in a field that required a symbolic address or a user symbol variable.
		150	A label was used in a field that required a symbolic address or a user symbol variable.
		154	Error in reading address for symbol entry.
		156	Illegal symbol in expression.
		159	Bad level specified in extended address. 1. More than 9 levels were specified in an extended address. 2. Something other than a ’(’ or a number followed a ’.’ in an extended address.
		160	Unrecognized section specifier. An illegal character followed the “$” in an address.
		161	Bad timer or counter specification 1. The first letter of the data table address is a T, C, or P, but there are not 4 characters in the specification. Incorrect addresses that would cause this error include C:15, $C5:3, CCUM:23, etc. 2. The key data table word specifier was 4 characters long and began with a T, C, or P, but it did not match the legal word specifiers (e.g., $TACM:3). 3. There was no colon following a legal word specifier.
		163	Missing colon between file and word.

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Meaning
		164	Illegal word specifier in a data table address.
		165	Illegal context specifier. When an expression determined the context in a data table address, or when the global context (context 0) was specified in a data table address, a colon followed the context.
		166	Attempt to execute a symbol not defined as a process. The system symbol exists but refers to a symbolic address rather than to a process.
		169	Either the number or the expression following the '/' in an address has a value outside the range 0 to 15 (decimal).
		171	Value specified in a bit assignment statement was other than a zero or a one.
		177	Illegal use of EXIT command.
		178	Illegal use of STOP command.
		179	STOP encountered in procedure.
		188	Attempt to read/write a bad address.
		189	Unable to evaluate the expression in the given base. This will occur for example, if the argument of a FROM...BCD function is not a valid BCD bit pattern. It will also occur when invalid characters occur in numeric values (e.g., “57 + 12X”).
		192	Function being used is not defined.
		194	Expression is too complex.
		199	Attempt to divide by zero.
		200	Bad port specifier. That is, the character following the '#' is other than 'H', 'h', 'M', or 'm'.
		201	User symbol used as part of remote address specification.
		202	Undefined data following assignment command. This error would occur, for instance, if the modifier UNRPOT were used instead of UNPROT.
		203	Error in remote specification. 1. A character other than '@' or '$' following the station num- ber specification (...+#H045*T...). 2. Something other than EOL, PROT, OR UNPROT following a remote source address (...=#H012$S5:8 + 9).
		204	Third-party transfer. That is, in an assignment command, both the source and the destination were remote addresses.
		205	Error in evaluating a PLC-2 address, or PLC-2 address greater than 65,535
		206	Zero range specified in an assignment command.

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Meaning
		207	Word range specified in destination address.
		208	Destination and source addresses disagree in type.
		209	Not of Data Highway message type.
		210	Use of a non-PLC-3 type address in a local address operand.
		211	In an assignment command, one of the local files does not exist, or the word specified is beyond the end of the file.
		213	A local file exists, but the action specified refers to addresses beyond the end of the file. Possible causes include: 1. In a word assignment statement, the offset is greater than the file size. 2. In a word range assignment statement, the sum of the base address and the offset is greater than the total file size. 3. In a file assignment statement, the destination file is smaller than the source file. If the source file is remote, a single packet will be fetched from the remote station's file.
		214	Local source and destination files differ in size
		215	The value resulting from operations specified on the left side of an assignment statement will not fit into the destination specified on the right side. 1. The source is in the H section and the destination is in the N section, but the number is too large (i.e., outside the range -32768 to +32767). 2. A word is transferred from a binary section (I, O, or B section) to the N or C section and the high-order bit is a 1. 3. The destination is in the D section, but the number is not a valid BCD bit pattern.
		217	More than 8 levels specified in file address.
		218	File size changed between packets of a multi-packet transaction.
		230	Reply packet too small

Reply Error Codes

The 1775-KA module inserts the reply error code in the STS or EXT STS bytes of any reply message packet it returns to a remote station. For reply errors, there is a direct correlation between the error codes in the STS and EXT STS bytes of reply messages and the error codes stored at the command station. Refer to section titled “PLC-3” (below “Transmissions Between Computer and Full-Duplex Modules”) for EXT STS bytes.

The meaning of each error code depends on the command message received from the remote station. The sections below described the error conditions that the various commands can generate. The error codes are listed according to the decimal value that would be stored at the command initiating station.

When a remote station transmits one of the commands listed below, the local 1775-KA module might issue a reply message that contains one of the error codes listed under that command. Error codes 81 to 88 appear in the STS byte of the reply message, and codes 231 to 241 appear in the EXT STS byte. PLC-3 and PLC family processors can only display error codes contained in the STS byte (80-89). Error codes contained in the EXT STS bytes are only available if either another PLC-3 or computer originates the command message.

Diagnostic Read Command

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
	10	81	1. A 2-byte ADDR field and a 1-byte SIZE field are missing after the FNC byte in the command message. 2. The number of bytes of data requested in the SIZE field is greater than the maximum number allowed per reply packet (244), or SIZE is 0 (zero).
	50	85	The command is an illegal request to read from the 1775-KA module's backplane window.

Diagnostic Status Command

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
	40	84	A backplane error occurred during determination of the physical address of the end of the ladder program or of the end of user memory.

PLC/PLC-2 Word Write Commands

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
	10	81	1. A 2-byte ADDR field is expected after the TNSW word, but only one byte is present. 2. There is an odd number of data bytes in the command packet. 3. The ADDR value is odd (that is, it does not specify a word address).
	30	83	The local 1775-KA module has executed a shutdown request to the local PLC-3 processor.
	40	84	Local PLC-3 backplane error (either memory parity or timeout/disconnect).
	50	85	1. The destination file does not exist in PLC-3 memory. 2. The destination word does not exist in the destination PLC-3 file.
	60	86	Local keyswitch setting prohibits writing into desired destination file.
	70	87	The local PLC-3 processor is in program mode. There may or may not be a major system fault.

PLC/PLC-2 Read Commands

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
	10	81	1. The required 2-byte ADDR field and 1-byte SIZE field are missing in the command message. 2. The ADDR value is odd (that is, it does not specify a word address). 3. The value of SIZE is 0 (zero). 4. The value of SIZE is greater than 244. 5. The SIZE value specifies an odd number of bytes.
	30	83	Same as for PLC/PLC-2 write commands above.
	40	84	Same as for PLC/PLC-2 write commands above.
	50	85	1. Destination file does not exist. 2. Destination file is too small. 3. Source file is more than 65,535 words long.
	70	87	PLC-3 processor is in program mode.

PLC/PLC-2 Bit Write Commands

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
	10	81	Incomplete bit description because the number of bytes after the TNSW is not a multiple of 4.
	30	83	Same as for PLC/PLC-2 word write commands above.
	40	84	Same as for PLC/PLC-2 word write commands above.
	50	85	1. Destination file does not exist 2. Destination bits do not exist in destination file 3. Length of source file is greater than 64, 535 words.
	60	86	Keyswitch setting at local PLC-3 processor prohibits access.
	70	87	Local PLC-3 processor is in program mode.

PLC-3 Write Commands

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
	10	81	1. There are not at least 2 bytes of data after the end of the block address. 2. There is an odd number of data bytes after the end of the block address. 3. Sum of packet offset and size values specifies more than 65,535 words. 4. Sum of packet offset and size is greater than total transaction size.
	30	83	The local 1775-KA module has executed a shutdown request.
	40	84	Backplane error (either memory parity or timeout/disconnect).
	60	86	Keyswitch setting disallows access.
	70	87	Local PLC-3 is in program mode.
1	F0	231	There is an error in converting the block address (major section >63, context >15, or section >15).
2	F0	232	Three or fewer addressing levels specified in for a PLC-3 word address.
3	F0	233	Conversion of a file address to block address resulted in more than 9 addressing levels.
4	F0	234	Symbolic address not found.
5	F0	235	Symbolic address is of length zero or is longer than 8 bytes.
6	F0	236	1. Destination file not found. 2. Destination address does not point to a word (for word-range writes) or a file (for file writes). 3. Destination address specifies more levels than required. 4. First word of destination location does not exist.
7	F0	237	1. Any word in the total transaction does not exist in the destination file. 2. For a file write, the source and destination files are not the same size.
8	F0	238	Destination file size changed between packets of a multi-packet transaction and became too small for the total transaction.
9	F0	239	There are more than 65,535 words in the source file.
A	F0	240	Sum of total transaction size and the word level of PLC-3 addressing is greater than 65,535.
B	F0	241	Source station does not have access to the destination file.

PLC-3 Read Commands

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
		81	1. There is more than one byte of data after the byte address. 2. Number of bytes to read is odd. 3. Number of bytes to read is zero. 4. Number of bytes to read is greater than the maximum allowed in a reply packet (244). 5. Sum of packet offset and size of data in words is greater than 65,535. 6. Sum of packet offset and size of data in words is greater than the total transaction size.
		83	The local 1775-KA module has executed a shutdown request.
		84	Backplane error (either memory parity or timeout/disconnect).
		87	Local PLC-3 is in program mode.
1	F0	231	Error in converting the block address (major section >63, context >15, section >15).
2	F0	232	Three or fewer addressing levels specified in for a PLC-3 word address.
3	F0	233	Conversion of a file address to block address resulted in more than 9 addressing levels.
4	F0	234	Symbolic address not found.
5	F0	235	Symbolic address is of length zero or is longer than 8 bytes.
6	F0	236	1. File not found. 2. Destination address does not have enough levels to specify a PLC-3 word (for word-range reads) or a file (for file reads). 3. The PLC-3 address specifies more levels than required. 4. Word specified by the PLC-3 address does not exist.
7	F0	237	1. Any of the destination words in the destination file do not exist. 2. For a file read, the source and destination files are not the same size.
8	F0	238	The file size decreased between packets of a multi-packet transaction and became too small for the total transaction.
9	F0	239	File is larger than 65,535 words.
A	F0	240	Sum of total transaction size and PLC-3 addressing is greater than 65,535.

PLC-3 Bit Write Commands

EXT STS Code (if applicable)	STS Code (if applicable)	Error Code	Possible Causes
	10	81	More than 4 bytes of data exist after the PLC-3 address in the command message.
	30	83	The local 1775-KA module has executed a shutdown request.
	40	84	Backplane error (memory parity or timeout/disconnect).
	60	86	Keyswitch setting disallows access.
	70	87	Local PLC-3 is in program mode.
1	F0	231	Error in converting the block address (major section >63, context >15, section >15).
2	F0	232	Three or fewer addressing levels specified in for a PLC-3 word address.
3	F0	233	Conversion of a file address to block address resulted in more than 9 addressing levels.
4	F0	234	Symbolic address not found.
5	F0	235	Symbolic address is of length zero or is longer than 8 bytes.
6	F0	236	1. File not found.
			2. Destination address does not specify a PLC-3 word.
			3. The PLC-3 address specifies more levels than required.
			4. Word specified by the PLC-3 address does not exist.
9	F0	239	File is larger than 65,535 words.
B	F0	241	Remote station does not have access to the destination file.

Remote Error Codes

Remote error codes are those reported in a reply message from a remote station that received a command message from the local PLC-3 station. These error codes are stored under user symbol ERROR in the local PLC-3 station.

The meaning of a particular remote error code will vary, depending on the type of communication interface module at the remote station. For example, if the remote station is a PLC-3 processor with a 1775-KA interface module, the remote error codes will have the meanings listed above. For the meanings of other remote error codes, refer to section titled “ERROR WORD in User Programming,” errors 80-89.

Internal Error Counter

These counters can be read through the diagnostic read command. They are available only to a device that can format the diagnostic commands. PC user program is unable to set up a diagnostic command. They are used to record events of interest for debugging new highway software and for longer term reliability analysis. The counters occupy a block of the internal scratch RAM. Most are single byte counters that wrap around to zero when they overflow.

These counters provide a useful tool for diagnosing problems. if a troubleshooter has available a device that will read the internal error counters of module, the ACK timeout counter and the false poll counter will be useful for diagnosing bad cabling or a noisy or loaded highway installation. Also useful is the ratio of messages transmitted (successfully returned an ACK from the remote station) versus the commands sent (but not necessarily completed).

NOTE: Because of differences in revision levels in a given module and variations from module to module, the user must first request the location of these counters by transmitting to the module a diagnostic status command. Then, based on the address returned, the number of the following counters can be used as an offset to calculate the location of the desired counter, or to calculate how many counter values he wants returned. This information is then used to format a diagnostic read command, and the reply from that will contain the data stored in the counters. Also, the counters are divided into two categories: Data Highway and RS-232, and they apply to activity on their respective parts. Obviously, the 1771-KA will only have Data Highway counters. Most modules will have both.

The second section lists internal error counters, bytes of information stored in RAM in each Data Highway module. These numbers can only be read by issuing a diagnostic read command from a device connected to one of the modules that supports an RS-232-C port.

1771-KA/1774-KA Data Highway Counters (only)

0. CRC error on ACK.

1. ACK timeout. Counts the number of times that the sender timed out waiting for an acknowledgement. This is a common error, and will be one of the first to respond to reflections or low level noise on the highway. It seems to be especially sensitive to problems on longer cables. It will also show up often if the receiver or transmitter circuitry on a module is marginal, or if the cable connections are loose.

2. Contention. Counts the number of times contention was detected. This will also show up quickly on noisy or overlength cables. This counter corresponds to error 93. If 93 is a common error on a highway system then expect 37 (start bit timeout) errors also, since any reply that experiences contention will not be re-tried.

3. Bad ACK status. Counts the number of times the ACK was successfully received but contained a nonzero status code other than memory full. Currently the only other implemented ACK code is buffer overflow. This condition should never occur except when debugging new code.

4. Returned messages. Counts the number of times the highway driver returns a message to sender. Each count corresponds to one local error bit set or one reply message lost.

5. Transmit: memory full. Counts the number of times that the receiving station's memory was full. Each time this happens the message is placed on a waiting queue for a half second. Each message will be re-tried five times for memory overflow before it is returned to sender.

6. Poll timeout. Counts the number of times that this station grabbed mastership of the highway because it timed out while waiting to hear a valid frame. On a highway that has just been powered up there should be only one station that has this counter incremented.

7. False poll. Counts the number of times that this station has tried to relinquish mastership and the station that was expected to take over failed to respond. This happens often on a noisy highway because the noise is mistaken for a poll response, and the wrong station is selected as the next master. When this occurs the old master resumes polling. It also can happen on a long highway if the poll response is very attenuated and is not picked up by the carrier detect circuit. If the new station does respond, but the old master does not hear it, the old master will record a false poll and continue polling. The new master will start polling also. This usually leads to the second station detecting contention and relinquishing.

8. Receiver heard status. Counts the number of times that the receiver received a status frame instead of a message frame. This should occur only if a poll timeout is imminent (a master has had mastership for more than 170 ms) and the station has disabled its address recognizer to test for any valid traffic.

9. Frame too small. Counts the number of frames that were rejected because the header was incomplete. This should only be counted because of undebugged software or in the unlikely event that a bad frame fooled the CRC checker.

10. Wrong destination address. Counts the number of frames that were rejected because the destination address was incorrect. This can have the same cause as #8. This counter also detects frames that have the same source and destination address.

11. Receiver: memory full. Counts the number of times that the receiver sent an ACK without first being able to allocate a receiver buffer. This will result in a memory overflow error when the next message is received.

12. Bad frame status. Counts the number of frames that were rejected because of a bad CRC. This error is very common on a noisy highway.

13. Buffer overflow. Counts the number of times a message was received that contained more than 250 bytes.

14. Memory overflow. Counts the number of times a message was received when there was not buffer space allocated for it. This usually follows a memory full error.

15. Retransmits. Counts the number of duplicate frames received. A duplicate frame is sent by a transmitter when it fails to receive an ACK. If the reason it failed to receive an ACK was that the ACK was lost, rather than because the original message was lost, the duplicate is redundant and should be discarded. Any two successive messages between polls that have the same sequence number fields and the same command/reply bit are assumed to be duplicates.

16. Aborts. Counts the number of aborts received. The HDLC abort signal is not used on the Data Highway, but can be detected by the SIO in certain circumstances. Some stations whose addresses match the ringing pattern after a transmitter shutoff can be particularly susceptible to this error (stations 36, 76, and 176 for example). These numbers will depend on highway configurations.

17. Transmitted messages. A 16-bit counter that records the number of messages successfully transmitted.

18.

19. Received messages. A 16-bit counter that records the number of messages successfully received.

20.

21. Commands send. A 16-bit counter that records the number of command messages that were successfully generated as the result of a start bit being set. Some of these messages may not be recorded as being transmitted either because they were not successfully sent or because they were sent to the same station that originated them.

22.

23. Messages executed. A 16-bit counter that records the number of command messages that were received to be executed from the highway. This count does not depend on whether execution was successful. For each message counted as received a reply message is sent.

24.

25. Replies received. A 16-bit counter that records the number of reply messages that were received that resulted in the setting of a done or remote error bit.

26.

27. Breaks. Counts the number of breaks sent to the IT.

28. Resyncs. Counts the number of times the PC driver has to resyncronize with the PC. This counter will always count at least one resync (because of powerup).

29. IT errors. Counts down modulo 5 the number of errors on the KA-to-IT cable. Every time this count reaches zero the KA does a handshake to reset the forced I/O table in the PC.

30. Undeliverable replies. Counts the number of replies that were lost because they could not be delivered over the highway. Undeliverable commands can be signaled to the user, because the "user" is located in PC memory, and can always be reached. If a reply message cannot be delivered over the highway there is no way to signal the user (of that message), who is also over the highway, that this station cannot signal a reply. The local user is not concerned with the problems of the remote user, and can take no meaningful action anyway, so there is not much to do but destroy the message and count it.

1771-KC

NOTE: The intelligent device can read the memory of the 1771-KC it is hooked to by setting the destination equal to the module address.

0. Bad CRC on ACK.

1. ACK timeout. Counts the number of times that the sender timed out waiting for an acknowledgement. This is a common error, and will be one of the first to respond to reflections or low level noise on the highway. It seems to be especially sensitive to problems on longer cables. It will also show up often if the cable connections are loose.

2. Contention. Counts the number of times contention was detected. This will also show up quickly on noisy or overlength cables. This counter corresponds to error 93.

3. Bad ACK status. Counts the number of times the ACK was successfully received but contained a nonzero status code other than memory full.

4. Returned messages. Counts the number of times the highway driver returns a message to sender with a nonzero status code because a reply was not received from a remote station. Each count corresponds to one local error bit set or one reply message lost.

5. Transmit: memory full. Counts the number of times that the receiving station's memory was full. Each time this happens the message is placed on a waiting queue for a half second. Each message will be re-tried five times for memory overflow before it is returned to sender.

6. Poll timeout. Counts the number of times this station grabbed mastership of the highway because it timed out while waiting to hear a valid frame. On a highway that has just been powered up there should be only one station that has this counter incremented.

7. False poll. Counts the number of times that this station has tried to relinquish mastership and the station that was expected to take over failed to respond. This happens often on a noisy highway because the noise is mistaken for a poll response, and the wrong station is selected as the next master. When this occurs the old master resumes polling. It also can happen on a long highway if the poll response is very attenuated and is not picked up by the carrier detect circuit. If the new station does respond but the old master does not hear it the old master will record a false poll and continue polling, and the new master will start polling also. This usually leads to the second station detecting contention and relinquishing.

8. Receiver heard status. Counts the number of times that the receiver received a status frame instead of a message frame. This should only occur if a poll timeout is imminent (a master has had mastership for more than 170 ms) and the station has disabled its address recognizer to test for any valid traffic. The probability of errors in #8, 9 and 10 increases substantially.

9. Frame too small. Counts the number of frames that were rejected because the header was incomplete. This should only be counted because of undebugged software or in the unlikely event that a bad frame fooled the CRC checker.

10. Wrong destination address. Counts the number of frames that were rejected because the destination address was incorrect. This can have the same cause as #8. This counter also detects frames that have the same source and destination address.

11. Receiver: memory full. Counts the number of times that the receiver sent an ACK without first being able to allocate a receive buffer. This will result in a memory overflow error when the next message is received.

12. Bad frame status. Counts the number of frames that were rejected because of a bad CRC. This error is very common on a noisy highway.

13. Buffer overflow. Counts the number of times a message was received that contained more than 250 bytes.

14. Memory overflow. Counts the number of times a message was received when there was no buffer space allocated for it. This usually follows a memory full error.

15. Retransmits. Counts the number of duplicate frames received. A duplicate frame is sent by a transmitter when it fails to receive an ACK. If the reason it failed to receive an ACK was that the ACK was lost, rather than that the original message was lost, the duplicate is redundant and should be discarded. Any two successive messages between polls that have the same sequence number fields and the same command/reply bit are assumed to be duplicates.

16. Aborts. Counts the number of aborts received. The HDLC abort signal is not used on the Data Highway, but can be detected by the SIO in certain circumstances. Some stations whose addresses match the ringing pattern after a transmitter shutoff can be particularly susceptible to this error (stations 36, 76, and 176 for example). These numbers will depend on highway configurations.

17. Transmitted messages. A 16-bit counter that records the number of messages successfully transmitted.

18.

19. Received messages. A 16-bit counter that records the number of messages successfully received.

20.

21. Number of ACKs received

22. No used

23. Number of ACKs

24. Not used

25. Number of NAKs received

26. Not used

27. Number of NAKs sent

28. Not used

29. Undeliverable replies. Counts the number of replies that were lost because they could not be delivered over the highway. Undeliverable commands can be signalled to the user, because the “user” is located in PC memory, and can always be reached. If a reply message cannot be delivered over the highway there is no way to signal the user (of that message), who is also over the highway, that this station cannot signal a reply. The local user is not concerned with the problems of the remote user, and can take no meaningful action anyway, so there is not much to do but destroy the message and count it.

30. Timeout preset

31. NAK preset. Values set by diagnostic commands or set by default on power up

32. ENQ preset

1771-KF Error Counters

DATA HIGHWAY COUNTERS

0. Bad CRC or I/O error on ACK. Same causes as bad CRC on messages.

1. ACK timeout. Counts the number of times that the sender timed out waiting for an acknowledgement. This is a common error, and will be one of the first to respond to reflections or low level noise on the highway. It seems to be especially sensitive to problems on longer cables. It will also appear often if the receiver or transmitter circuitry on a module is marginal, or if the cable connections are loose.

2. Contentions. Counts the number of times contention was detected. This will also appears quickly on noisy or overlength cables. This counter corresponds to error 93.

3. Bad ACK status. Counts the number of times the ACK was successfully received but contained a nonzero status code other than memory full. Currently the only other implemented ACK code is buffer overflow. This condition should never occur except possibly when debugging new computer programs.

4. Returned messages. Counts the number of times the highway driver returns a message to sender. Each count corresponds to one local error bit set or one reply message lost.

5. Transmit: memory full. Counts the number of times the highway driver returns a message to sender. Each time this happens the message is placed on a waiting queue for a half second. Each message will be re-tried five times before it is returned to the sender.

6. Poll timeout. Counts the number of times this station grabbed mastership of the highway because it timed out while waiting to hear a valid frame. On a highway that has just been powered up there should be only one station that has this counter incremented.

7. False poll. Counts the number of times that this station has tried to relinquish mastership and the station that was expected to take over failed to respond. This happens often on a noisy highway because the noise is mistaken for a poll response, and the wrong station is selected as the next master. When this occurs the old master resumes polling. It also can happen on a long highway if the poll response is very attenuated and is not picked up by the carrier detect circuit. If the new station does respond but the old master does not hear it the old master will record a false poll and continue polling, and the new master will start polling also. This usually leads to the second station detecting contention and relinquishing.

8. Receiver heard status. Counts the number of times that the receiver received a status frame instead of a message frame. (This counter will never be incremented because the message size is checked first, and all status messages are too small to be accepted.

9. Frame too small. Counts the number of frames that were rejected because they were less than 6 bytes long. This counter will record all status frames that were received by a station that disabled its address recognizer in the second step of the mastership timeout process. This will happen often on heavily loaded highway.

10. Wrong destination address. Counts the number of frames that were rejected because the destination address was incorrect. This can have the same cause as #9. This counter also detects frames that have the same source and destination address.

11. Receiver: memory full. Counts the number of times that the receiver sent an ACK without first being able to allocate a receiver buffer. This may result in a memory overflow error when the next message is received.

12. Bad frame status. Counts the number of frames that were rejected because of a bad CRC. This error is very common on a noisy highway.

13. Buffer overflow. Counts the number of times a message was received that contained more than 250 bytes.

14. Memory overflow. Counts the number of times a message was received when there was no buffer space allocated for it. This usually follows a memory full error.

15. Retransmits. Counts the number of duplicate frames received. A duplicate frame is sent by a transmitter when it fails to receive an ACK. If the reason it failed to receive an ACK was that the ACK was lost, rather than because the original message was lost, the duplicate is redundant and should be discarded. Any two successive messages between polls that have the same sequence number fields and the same command/reply bit are assumed to be duplicates.

16. Aborts. Counts the number of aborts received. The HDLC abort signal is not used on the Data Highway, but can be detected by the SIO in certain circumstances. Some stations whose addresses match the ringing pattern after a transmitter shutoff can be particularly susceptible to this error (stations 36, 76, and 176 for example). These numbers will depend on highway configurations.

17. Transmitted messages. A 16-bit counter that records the number of messages successfully transmitted.

18.

19. Received messages. A 16-bit counter that records the number of messages successfully received.

20.

MODEM CHANNEL COUNTERS

21. 16-bit count of the number of times the station attempted to send a message.

22.

23. 16-bit count of the number of messages that were successfully transmitted and ACKed.

24.

25. 16-bit count of the number of ACKs that were received.

26.

27. Number of ACKs successfully passed from the receiver's separator to the transmitter.

28. Number of NAKs received.

29. Number of NAKs passed from the separator to the transmitter.

30. Number of timeouts waiting for a response.

31. Number of ENQs sent.

32. Number of messages that could not be successfully sent.

33. Number of reply messages that could not be forwarded and which were destroyed.

34. 16-bit count of messages received.

35.

36. 16-bit count of ACKs sent.

37.

38. Number of NAKs sent.

39. Number of ENQs received.

40. Number of retransmissions received and ACKed. Retransmission is a message that has a transparent word, command, and source that match the previous message.

41. Number of STX (full-duplex mode) or SOH (half-duplex mode) received. This is in effect a count of the number of messages that were started.

42. Number of messages, characters, or message fragments that were ignored.

43. Number of messages that were aborted by receipt of a DLE ENQ.

44. Number of messages that were aborted by the receipt of an unexpected control code other than DLE ENQ.

45. Number of times the DLE ACK response was sent but no buffer space for the next message.

46. Number of times DLE NAK was sent because there was no buffer.

47. Number of broadcast messages received.

48. Number of broadcast messages that were successfully received.

49. Number of messages seen that were not for this station.

50. Number of DLE EOTs sent.

51. Number of calls received.

52. Number of times that phone was hung up by the module.

53. Number of times that DCD was lost.

54. Number of times that the phone was hung up because of a DCD timeout.

1771-KG Error Counters

MODEM CHANNEL COUNTERS

0. 16-bit count of the number of times the station attempted to send a message.

1.

2. 16-bit count of the number of messages that were successfully transmitted and ACKed.

3.

4. 16-bit count of the number of ACKs that were received.

5.

6. Number of ACKs successfully passed from the receiver's separator to the transmitted.

7. Number of NAKs received

8. Number of NAKs passed from the separator to the transmitter.

9. Number of timeouts waiting for a response.

10. Number of ENQs sent.

11. Number of messages that could not be successfully sent.

12. Number of reply messages that could not be forwarded and which were destroyed.

13. 16-bit count of messages received.

14.

15. 16-bit count of ACKs sent.

16.

17. Number of NAKs sent.

18. Number of ENQs received.

19. Number of retransmissions received and ACKed. A retransmission is a message that has a transparent word, command, and source that match the previous message.

20. Number of STX (full-duplex mode) or SOH (half-duplex mode) received. This is in effect a count of the number of messages that were started.

21. Number of messages, characters, or message fragments that were ignored.

22. Number of messages that were aborted by receipt of a DLE ENQ.

23. Number of messages that were aborted by the receipt of an unexpected control code other than DLE ENQ.

24. Number of times the DLE ACK response was delayed because of a lack of buffer space for the next message.

25. Number of times the reply was changed from ACK to NAK because unexpected characters (any besides DLE ENQ) were received while waiting for memory to free up.

26. Number of broadcast messages received.

27. Number of broadcast messages that were successfully received.

28. Number of messages seen that were not for this station.

29. Number of poll messages received for this station.

30. Number of DLE EOTs sent.

31. Number of calls received.

32. Number of times that phone was hung up by the module.

33. Number of times that DCD was lost.

34. Number of times that the phone was hung up because of a DCD timeout.

INTERVAL EVENT COUNTERS

35. Number of messages routed to RS-232 port.

36. Number of commands routed to command executor.

37. Number of replies routed to reply processor.

38. Number of messages sent to self.

39. Number of routing errors on inbound messages.

40. Number of routing errors on outbound messages.

41. Number of messages with incorrect network address.

42. 16-bit count of messages sent by command initiator.

43.

44. 16-bit count of commands received by command executor.

45.

46. 16-bit count of replies sent by command executor.

47.

48. 16-bit count of replies received by command initiator.

49.

50. Number of breaks sent to IT.

51. Number of resyncs sent to PC.

1775-KA Diagnostic Counters

DATA HIGHWAY PORT

1. Bad CRC on acknowledgement (Local error “A”)

2. No acknowledgement before timeout occurred (Local error “B”)

3. Contention (while master, detected message transmission by another station)

4. Acknowledgement contained an error (Local error “C”D)

5. Local errors (Sum of A, B, and C above)

6. Waits (no receive buffer space at destination station)

7. Timed out (master failed)

8. False polls (failure to transfer)

9. Received acknowledgment when not master

10. Message size too small (less than 5 bytes)

11. Incorrect DST, or SRC = DST

12. Memory not available for receive buffer

13. Received message has bad CRC value

14. Message too long

15. Message arrived when no buffer space left

16. Retransmissions of previously received message

17. Aborts (result of line noise)

18. Messages successfully transmitted

19. Messages successfully transmitted

20. Messages successfully received

21. Messages successfully received

22. Command messages sent

23. Command messages sent

24. Reply messages received

25. Reply messages received

26. Command messages received

27. Command messages received

28. Reply messages sent

29. Reply messages sent

MODEM PORT

1. Command messages sent

2. Command messages sent

3. Reply messages received

4. Reply messages received

5. Command messages received

6. Command messages received

7. Reply messages sent

8. Reply messages sent

9. ACKs received

10. ACKs received

11. ACKs sent

12. ACKs sent

13. NAKs received

14. NAKs received

15. NAKs sent

16. NAKs sent

17. Undeliverable reply messages

18. Computer link timeout (preset to 500 msec)

19. Computer link timeout (preset to 500 msec)

20. Maximum number of NAKs accepted per message (preset to 10)

21. Maximum number of ENQs sent per message (preset to 10)

22. Current NAK count

23. Current ENQ count

Transmissions Between Computer and Full-Duplex Modules

PLC-2/PLC

The full-duplex protocol (and its half-duplex variation) provides for a message packet that contains a reply to have a status byte reserved in its header. This byte (known as the STS byte) provides information about the execution or failure of the corresponding command that was transmitted from the computer. A reply that returns zeros in the STS byte means the command was executed at the remote station. Non-zero status can be divided into two categories: remote errors and local errors. Remote errors mean that a command was successfully transmitted by the Data Highway controller to another Data Highway station, but the remote station was unable to execute the command. The remote station then formatted a reply with the STS byte containing some error code. Local errors mean that the Data Highway controller was unable to transmit the message to the remote station. The local station then turns the command around, stuffs the STS byte with the appropriate error code, and returns it to the sender (computer). The error code format is as follows:

When the error is local, the high nibble (7-4) is zero, and the reference to a particular code is in the low nibble as a Hex value. When the error is remote, the low nibble is zero and the error code is in the high nibble. Since the full-duplex message packet is actually transmitted on the highway itself (in a encapsulated form), the contents of the STS byte is what other (PC) highway stations use to generate the error word in the PC application program. To decode the contents of the STS byte, refer in section titled ERROR WORD in User Programming to error codes (80-89) for remote errors and error codes (90-93) for local errors.

For remote station errors, make sure the error codes correspond to the processor type at that station (either PLC/PLC-2 or PLC-3).

Match the last digit of the error code with value in hex in the appropriate nibble. For example, if the STS byte contains 04H, the low nibble=4 and the high nibble is 0, indicating a local error. Referring to the local error codes (90-01), 04H matches to error 94 (contention on the highway).

PLC-3

NOTE: With PLC-2 compatibility mode for the 1775-KA, PLC-2 level commands addressed to a PLC-3 will only return errors as described above, that is, in the STS byte format.

In addition to the above, PLC-3 can also create a second layer of error codes, relative to PLC-3 type commands (CMD byte = 15). If the command is a PLC-3 level command addressed to a remote PLC-3, then the remote error returned from the 1775-KA will have an additional status byte stuffed into the data area called an EXT STS.

If there is a non-zero error value in the EXT STS byte, the STS byte will contain FOH (which, in effect, functions as a flag that there has been an error value generated). If the STS byte is zero, then the EXT STS will also be zero.

The following is a listing of relevant PLC-3 status codes for both the STS byte and the EXT STS byte:

Bits	Hex Value	Meaning
04 to 07 in STS byte	0	No error
	1	1. Illegal command or command size 2. Specified data size (number of bytes) is zero, odd, or greater than 512.
	4	PLC-3 backplane fault occurred during message execution.
	5	1. Read/write file does not exist. 2. PLC-2 addressing violation (read/write file too small) 3. Read/write file overflow (more than 65,535 words) 4. Diagnostic read command attempted to read the PLC-3 backplane window 5. Invalid physical address 6. Attempted to write data past the end of memory 7. Attempted to read data from beyond the end of user program memory.
	6	1. Memory protect keyswitch disallows access into file 2. Upload/download option is not enabled at the destination station
	7	PLC-3 in Program Mode
	8	No file assigned to source station
04 to 07 in STS Byte	F	Extended error format - look in the EXT STS byte for the error code
EXT STS byte	1	Error in conversion of block address
	2	Improper format for PLC-3 word address
	3	Error in conversion of file address
	4	Invalid symbol
	5	Improper symbol specification format
	6	Invalid PLC-3 word address
	7	Improper file size
	8	File size changed during message execution
	9	File size too large
	A	Message size too large
	B	Write privileges not granted to remote station
	C	Upload/download access is not available
	D	Requesting station already has upload/download access privileges

Bits	Hex Value	Meaning
	E	Shutdown request could not be executed
	F	Requesting station does not have upload/download access privileges.

Switch Settings

Switch Settings

Here is a reference for switch settings for a 1771–KE/KF communication controller module. (Prior to revision H.)

Switch Assembly SW–1

Switch 1:	Off– Full Duplex
	On – Half duplex
Switch 2:	Off– Embedded response disabled
	On – Embedded response enabled
Switch 3:	Off– Duplicate message detection disabled
	On – Duplicate message detection enabled
Switch 4:	Off– RS–232–C handshaking disabled
	On – RS–232–C handshaking enabled
Switch 5:	Off– Remote diagnostics pass through
	On – Remote diagnostics enabled

Here is a reference for switch settings for a 1771–KE/KF communication controller module. (Revision H.)

If you want to select protocol as:	**with error check as:**	**with parity as:**	**with embedded responses:**	**SW–1**				
				1	**2**	**3**	**4**	**5**
full duplex	BCC	none	no	off	off	N/A	N/A	off
full duplex	BCC	even	no	on	off	↓	↓	off
full duplex	BCC	none	yes	off	on			off
full duplex	BCC	even	yes	on	on			off
half duplex	BCC	none	no	off	off			on
half duplex	BCC	even	no	on	off			on
full duplex	CRC	none	yes	off	on			on
half duplex	CRC	none	no	on	on			on

If you want the module to:	**set switch 3:**
use handshaking signals	on
ignore handshaking signals	off

If you want the module's RS–232–C port to:	**set switch 4:**
detect and ignore duplicate messages	on
accept all messages regardless of duplication	off

A. First Digit Station (SW–2, SW–3, SW–4)

Switch		Digit
1	2	
Off	Off	0
Off	On	1
On	Off	2
On	On	3

B. Second and Third Digits

Switch			Digit
1	2	3	
Off	Off	Off	0
Off	Off	On	1
Off	On	Off	2
Off	On	On	3
On	Off	Off	4
On	Off	On	5
On	On	Off	6
On	On	On	7

Data Highway Baud Rate (SW–5)

Baud Rate	Switch	
	1	2
38,400	Off	Off
57,600 (recommended)	On	On
76,800	On	Off
115,200	Off	On

Computer Link Baud Rate and Parity (SW–6)

Baud Rate	Switch		
	1	2	3
110	Off	Off	Off
300	On	Off	Off
600	Off	On	Off
1200	On	On	Off
2400	Off	Off	On
9600	Off	On	On
19,200	On	On	On

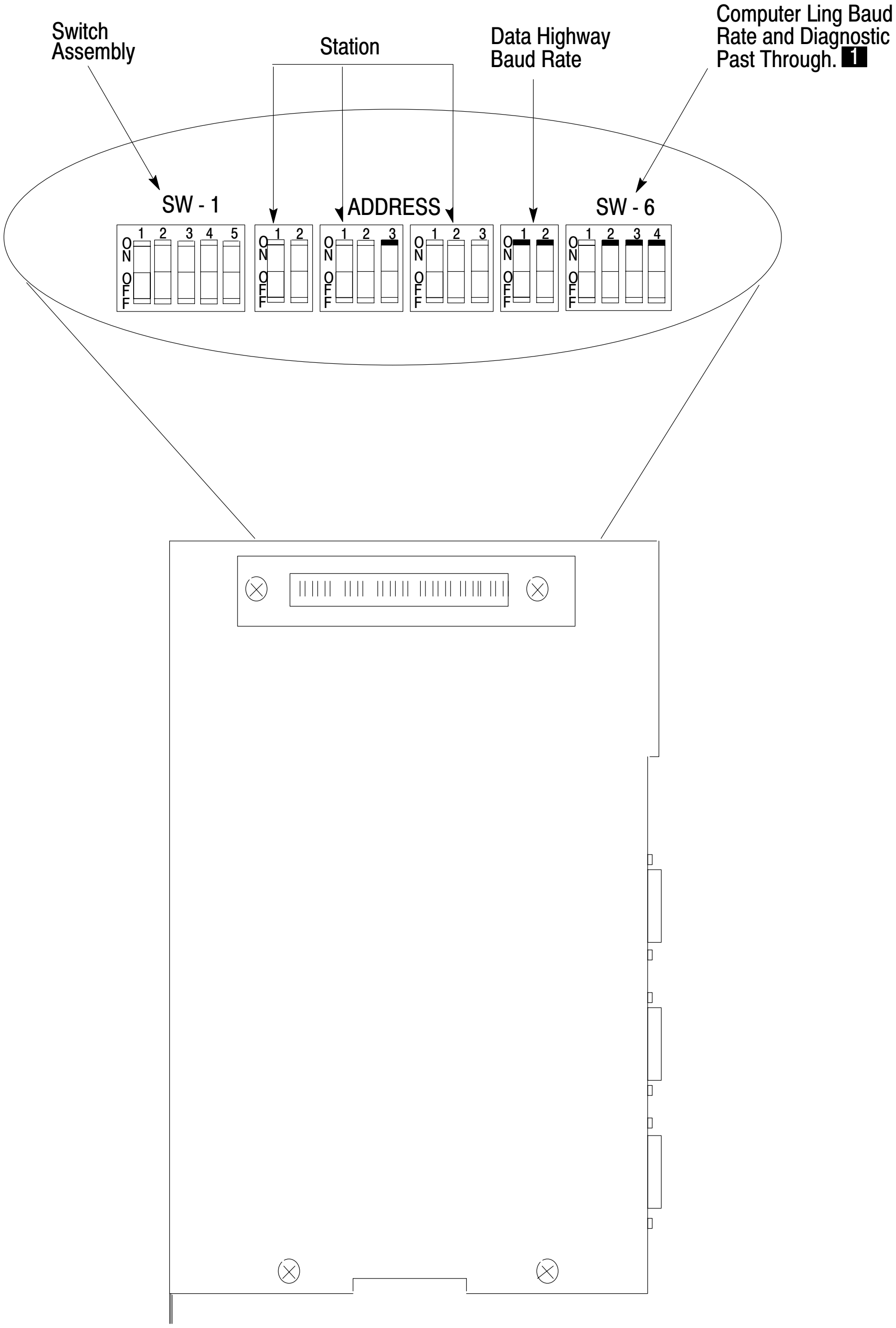
If you have revisions A–G module:

Switch 4: On–Even parity
Off – No

If you have revision H module:

Switch 4: On – Execute diagnostic command
Off – Pass through diagnostic command

Figure A.1
1771-KE,-KF Communication Controller Module



[1] Parity prior to series A revision H.

Appendix B

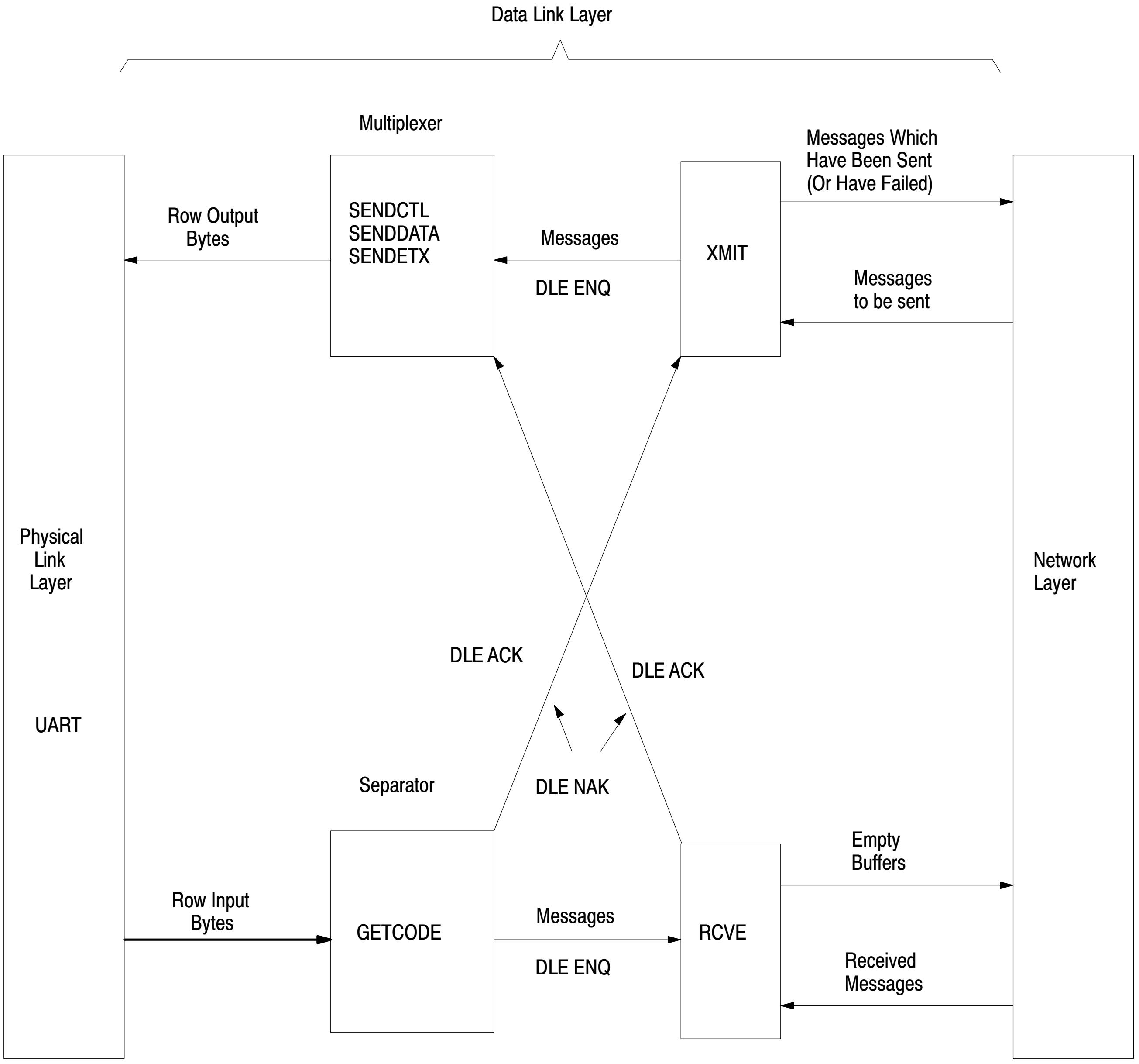
Detailed Flow Charts

General

This appendix presents flow charts that give **detailed** views of an example of software logic for implementing full–duplex protocol. The flow charts in chapter 4 gave you a **simplified** view of this software logic.

We have not shown any error checking or recovery relating to interaction with the modem handshake drive, a third process. To do this would overly complicate the flow charts, and in many cases, such error checking and recovery are not needed.

Figure B.1
Data Flow Diagram for Full–Duplex Protocol

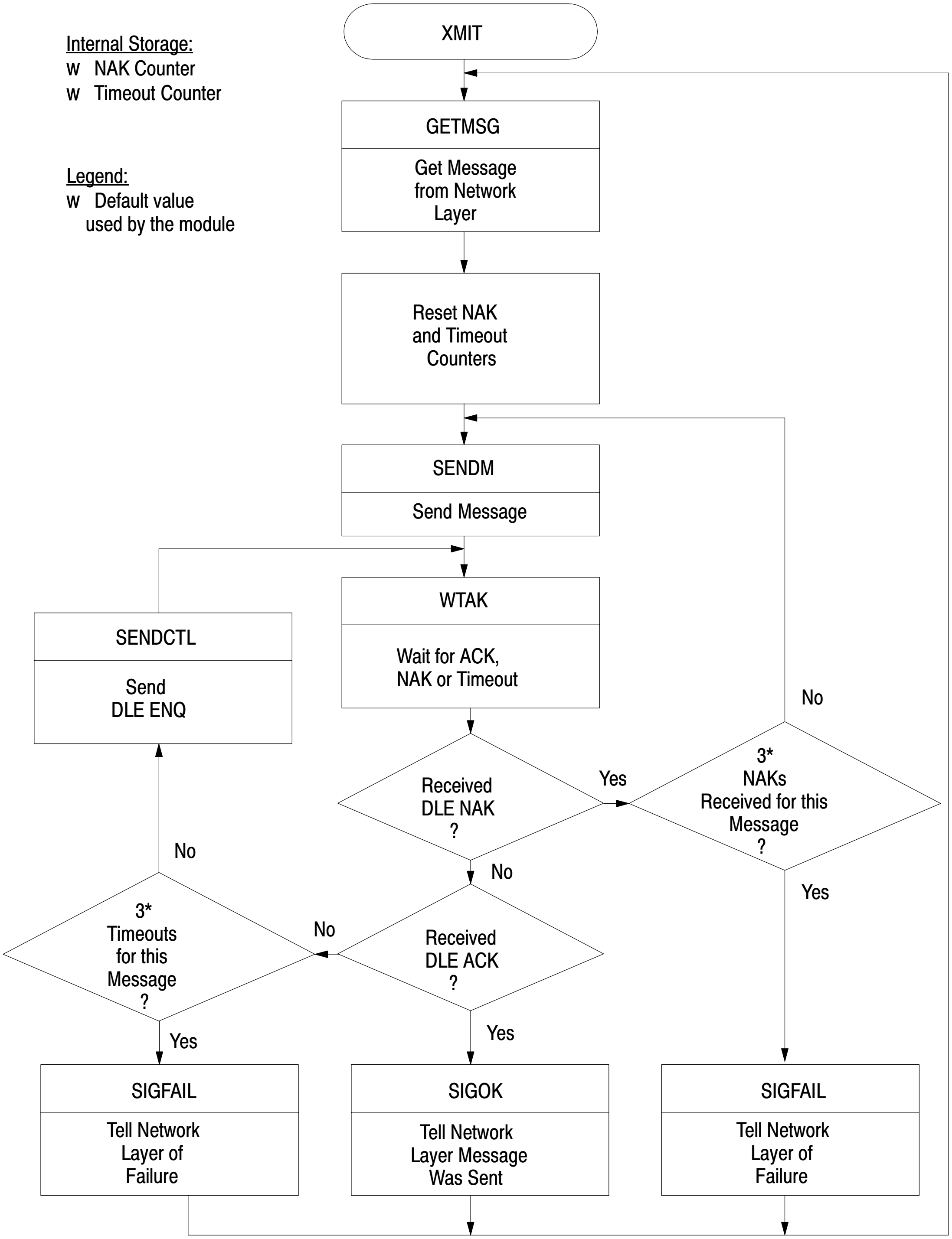


11163–I

Figure B.2
Transmitter Routine for Full Duplex Protocol

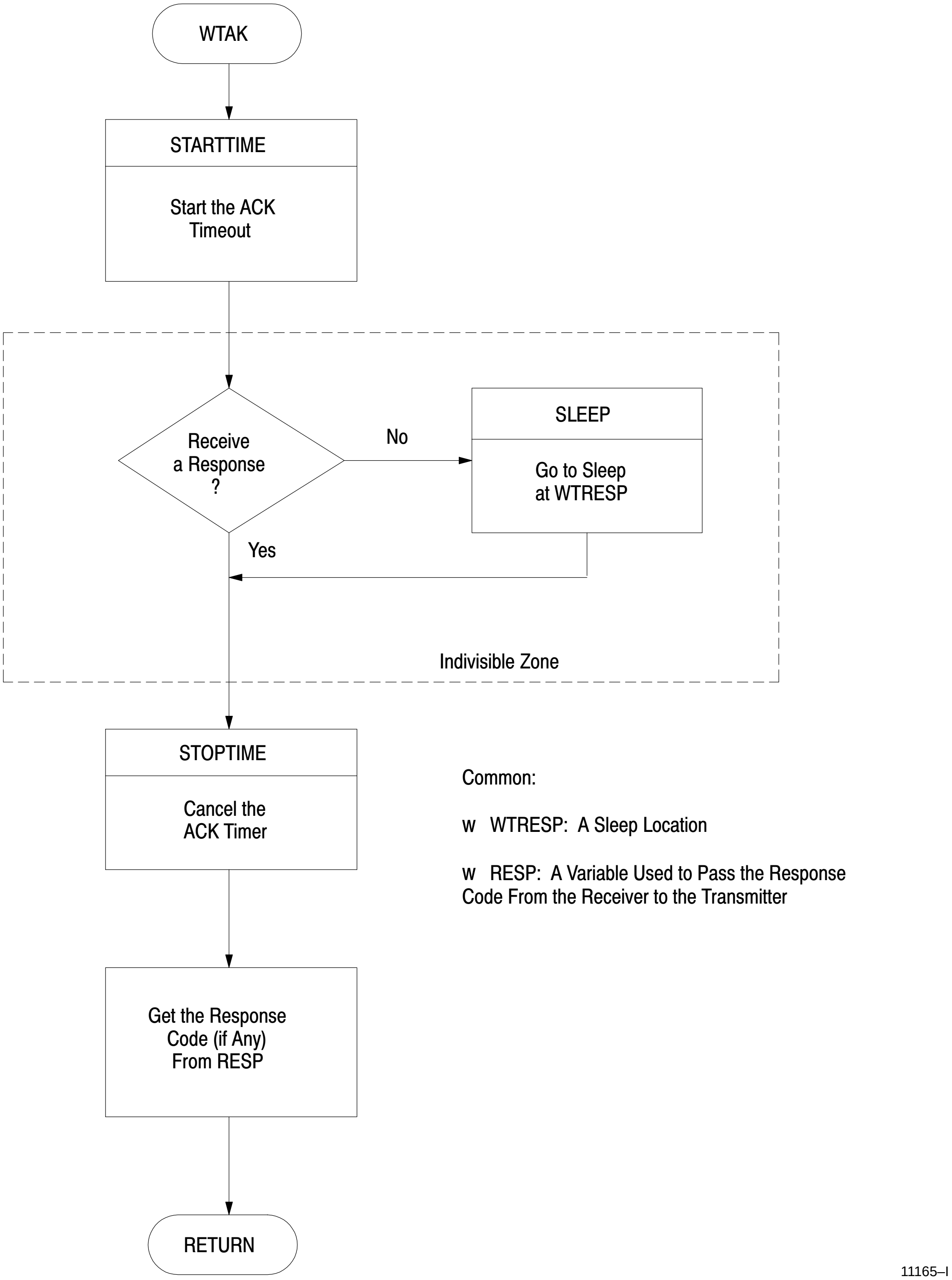
Internal Storage:
w NAK Counter
w Timeout Counter

Legend:
w Default value
used by the module


XMIT
GETMSG
Get Message from Network Layer
Reset NAK and Timeout Counters
SENDM
Send Message
WTAK
Wait for ACK, NAK or Timeout
SENDCTL
Send DLE ENQ
Received DLE NAK ?
Yes
No
3* NAKs Received for this Message ?
No
Yes
Received DLE ACK ?
No
Yes
3* Timeouts for this Message ?
No
Yes
SIGFAIL
Tell Network Layer of Failure
SIGOK
Tell Network Layer Message Was Sent
SIGFAIL
Tell Network Layer of Failure

11164–I

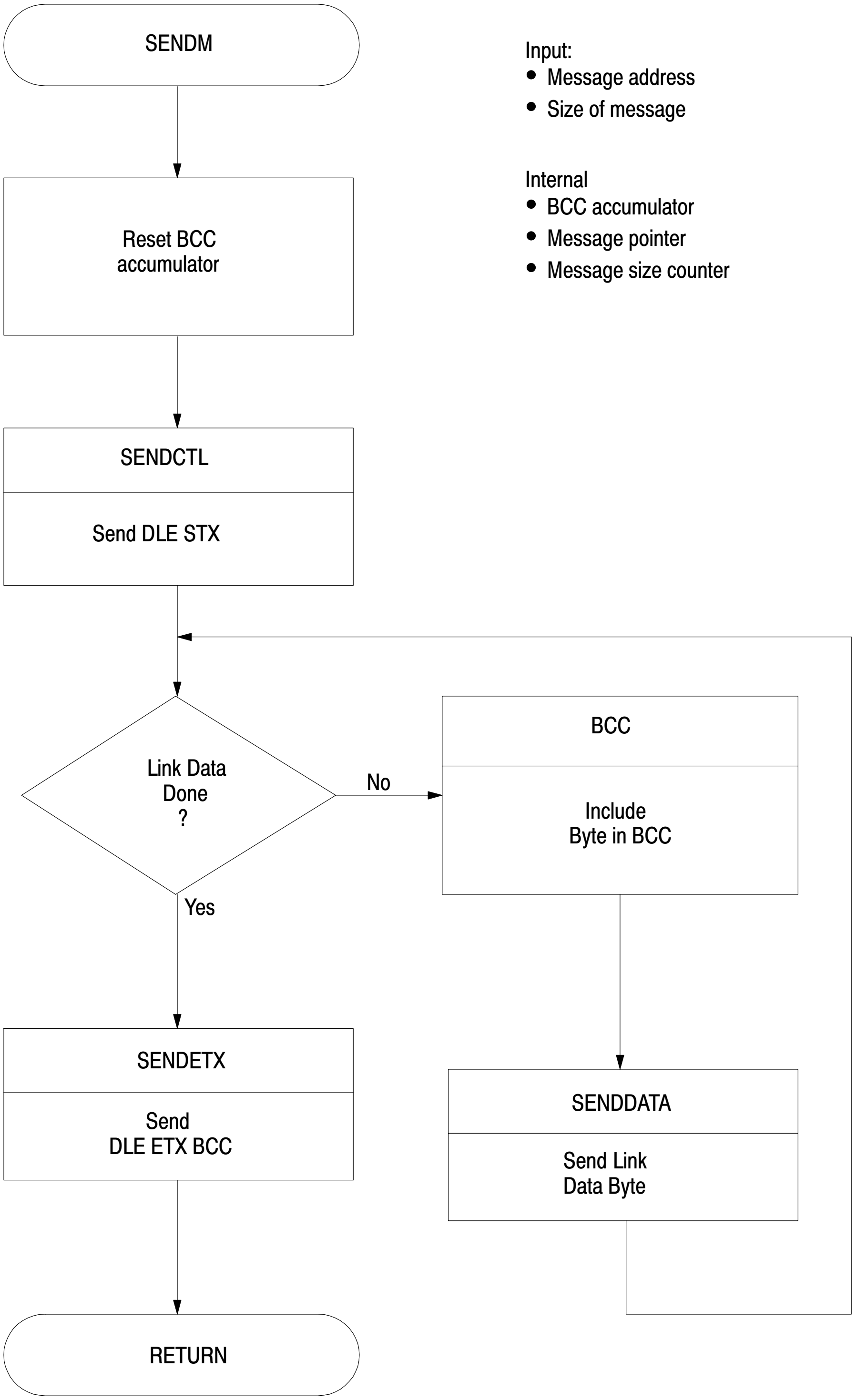
Figure B.3
WTAK Subroutine



Common:

- WTRESP: A Sleep Location
- RESP: A Variable Used to Pass the Response Code From the Receiver to the Transmitter

Figure B.4
SENDM Subroutine



Input:
- Message address
- Size of message

Internal
- BCC accumulator
- Message pointer
- Message size counter

11166–I

Figure B.5
STARTTIME Subroutine

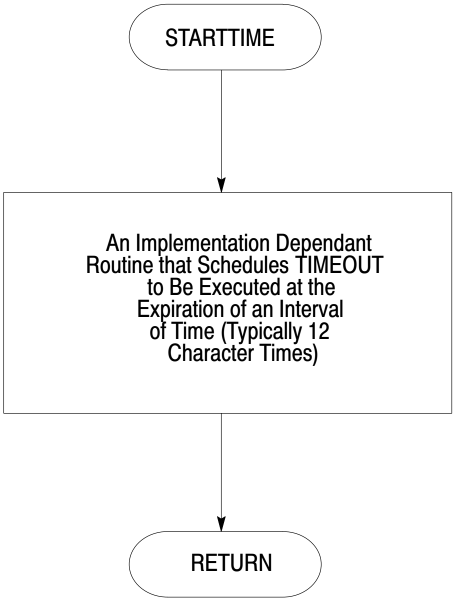

STARTTIME
An Implementation Dependant Routine that Schedules TIMEOUT to Be Executed at the Expiration of an Interval of Time (Typically 12 Character Times)
RETURN
11167–I

Figure B.6
STOPTIME Subroutine

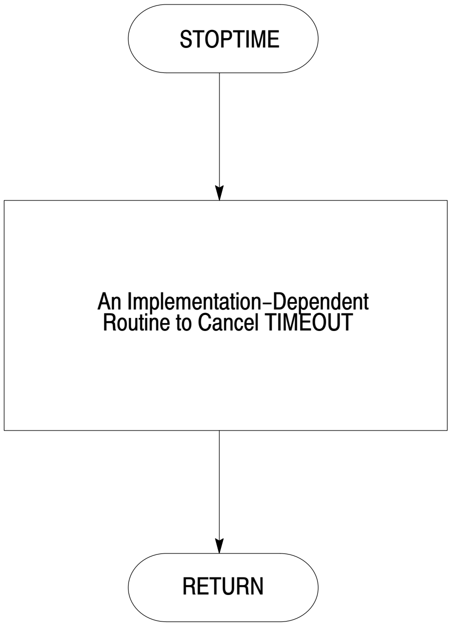
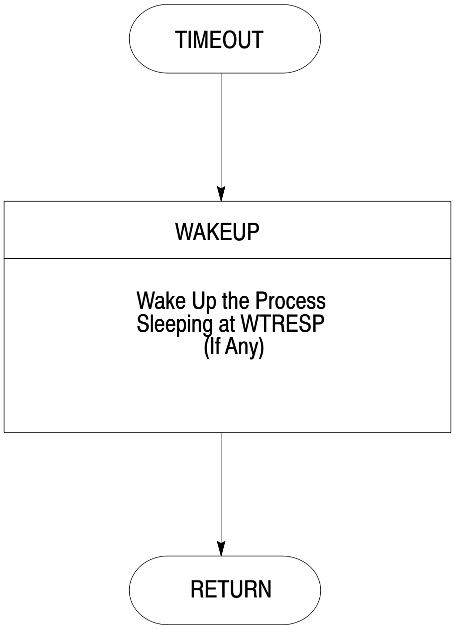


Figure B.7
TIMEOUT Subroutine



Scheduled By:

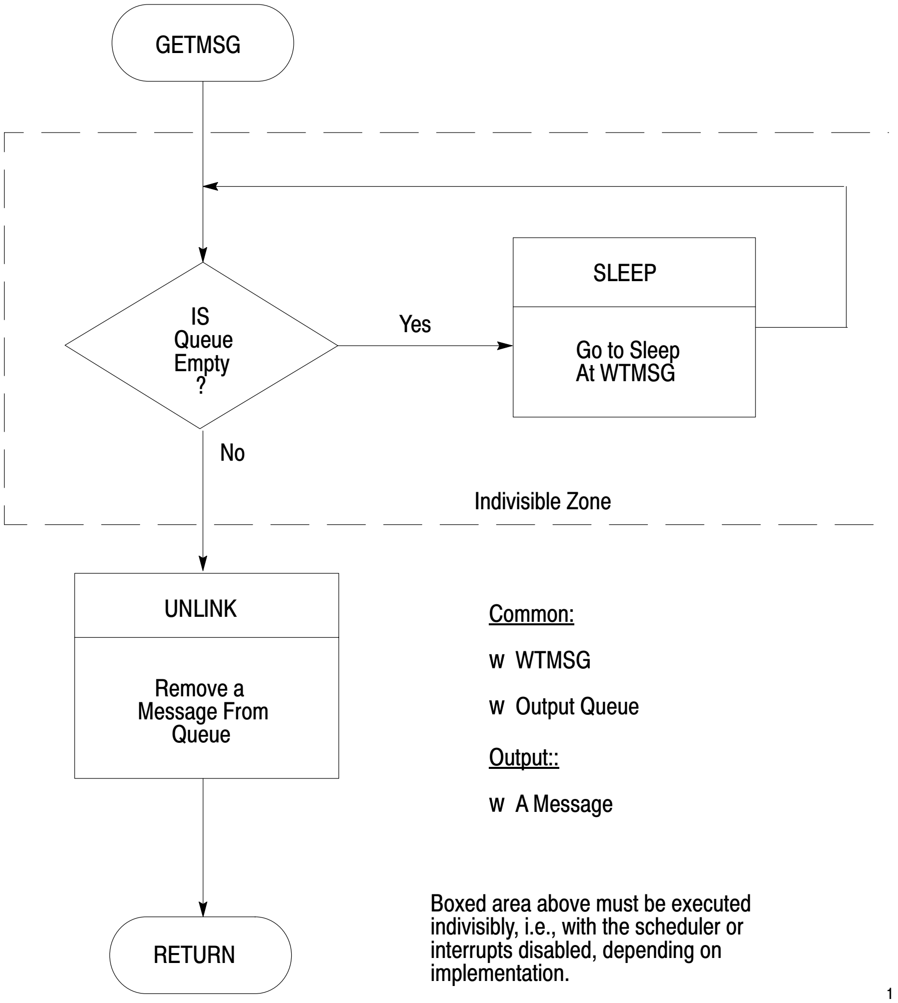
- STARTTIME

Aborted By:

- STOPTIME

11169–I

Figure B.8
GETMSG Subroutine



Common:

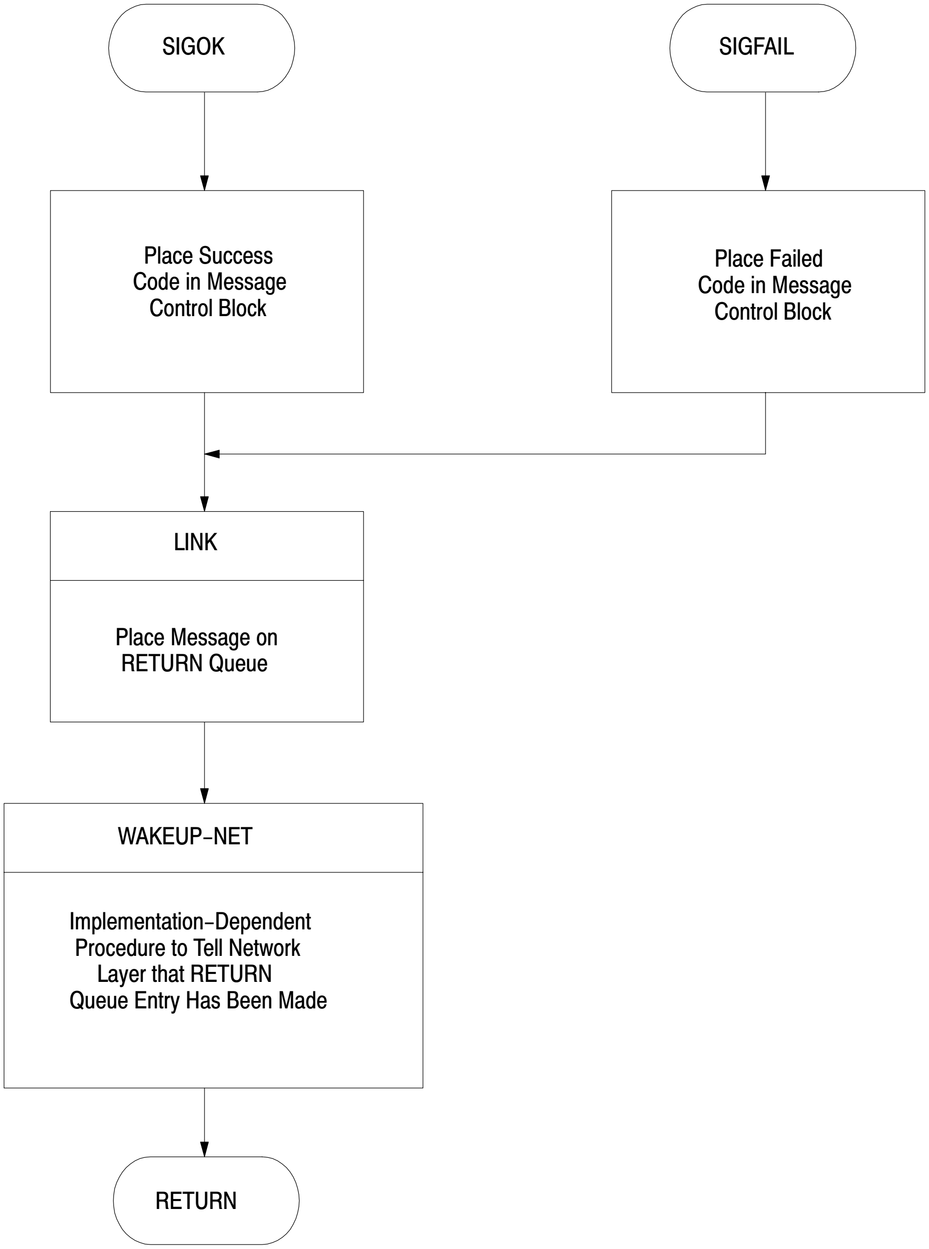
- WTMSG
- Output Queue

Output::

- A Message

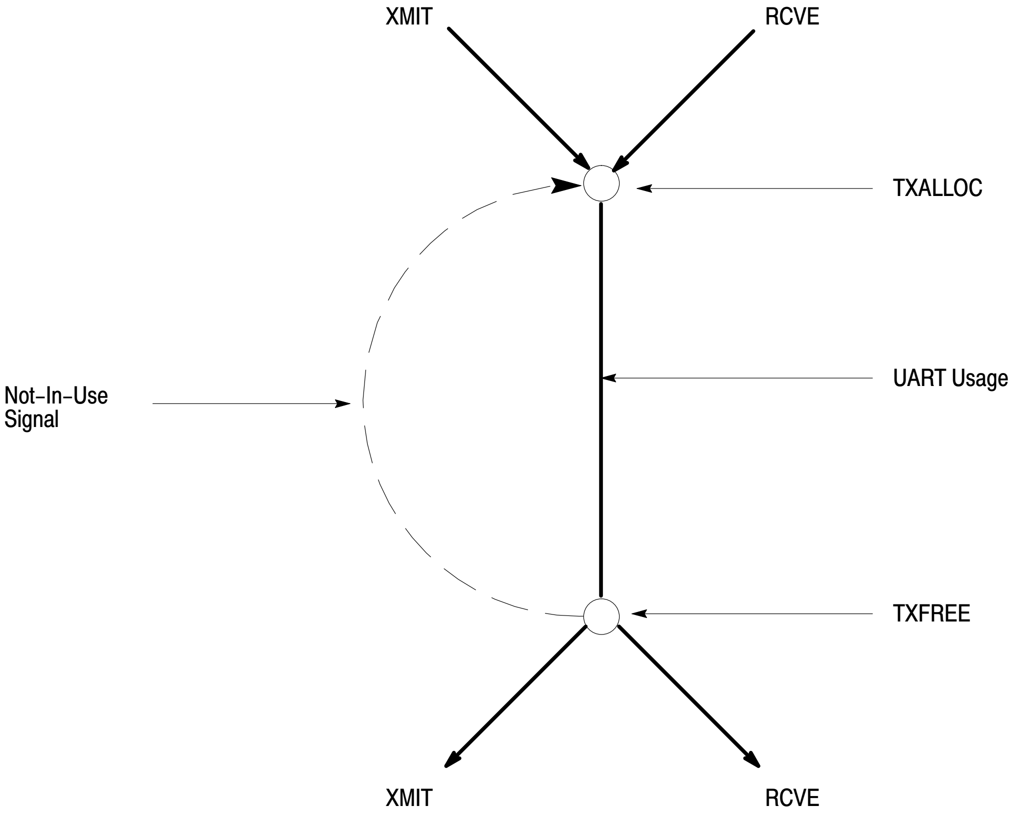
Boxed area above must be executed indivisibly, i.e., with the scheduler or interrupts disabled, depending on implementation.

Figure B.9
SIGOK/SIGFAIL Subroutine


SIGOK
SIGFAIL
Place Success Code in Message Control Block
Place Failed Code in Message Control Block
LINK
Place Message on RETURN Queue
WAKEUP–NET
Implementation–Dependent Procedure to Tell Network Layer that RETURN Queue Entry Has Been Made
RETURN

11171–I

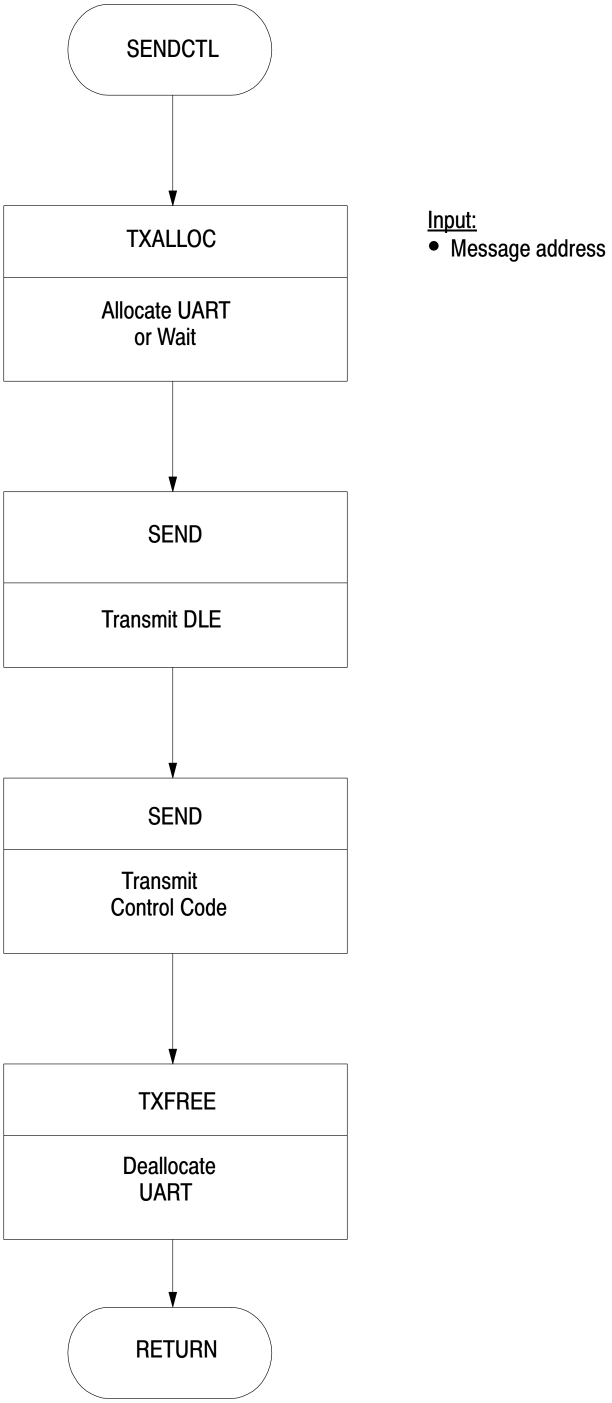
Figure B.10
Sharing the Transmit Side of the UART



Since the tramsit side of the UART is shared by the transmit and receive processes, a mutual exclusion mechanism is used to gain sole access during transmission of each indivisible code sequence. The UART is like a small bridge, and the mutual exclusion mechanism is like two flagmen, one at each end of the bridge. If two heavy trucks (one called XMIT and one called RCVE) come to the bridge at the same time, the flagman on that side lets one through and makes the other wait. When the first truck has crossed the bridge, the flagman on the far side signals the first flagman, who then allows the second truck to cross. In a similar manner, TXALLOC and TXFREE work together to ensure that XMIT and RCVE do not try to use the UART at the same time.

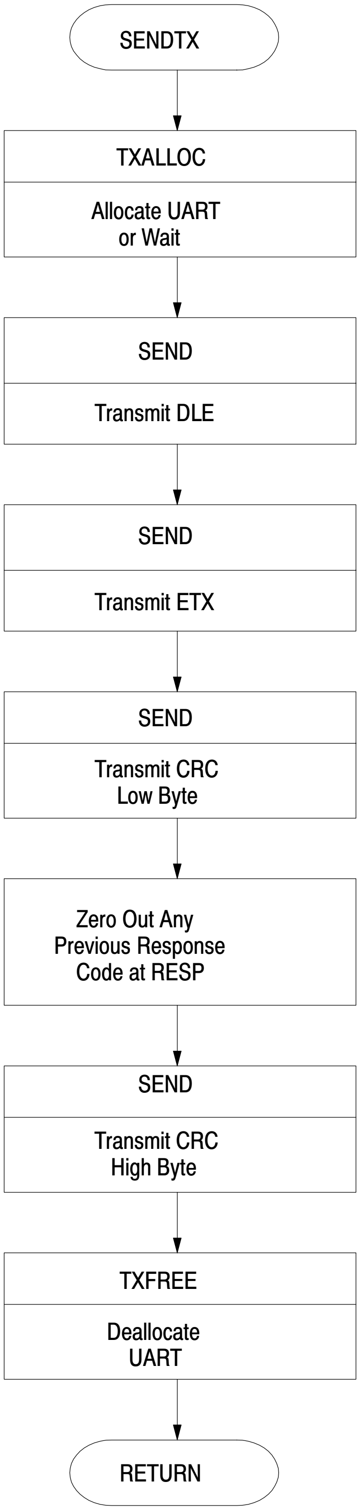
11172–I

Figure B.11
SENDCTL Subroutine



11173–I

Figure B.12
SENDTX Subroutine



Input:

- BCC

Common:

- RESP: The Response Code Variable

11174–I

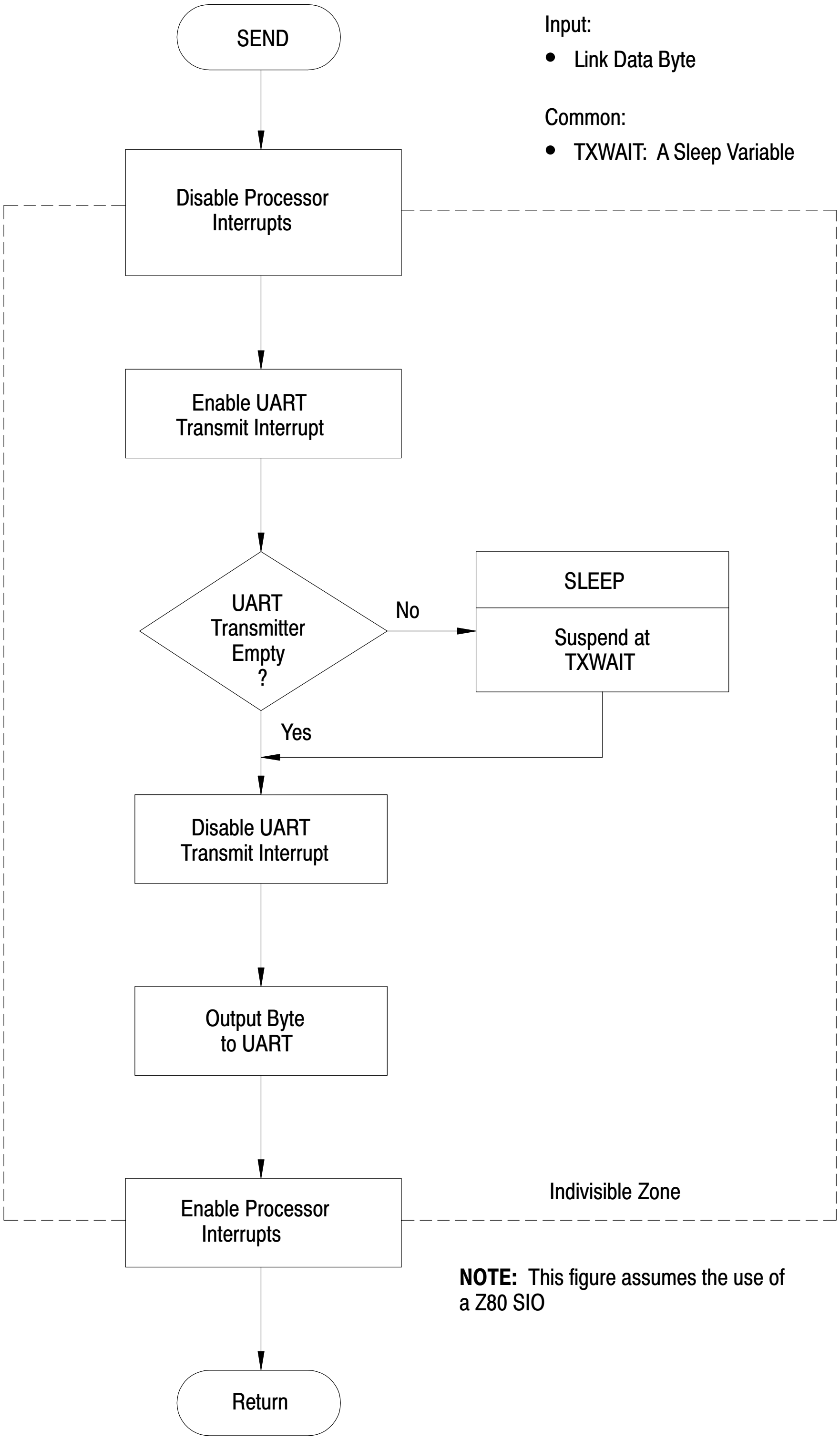
Figure B.13
SEND Subroutine

Input:

- Link Data Byte

Common:

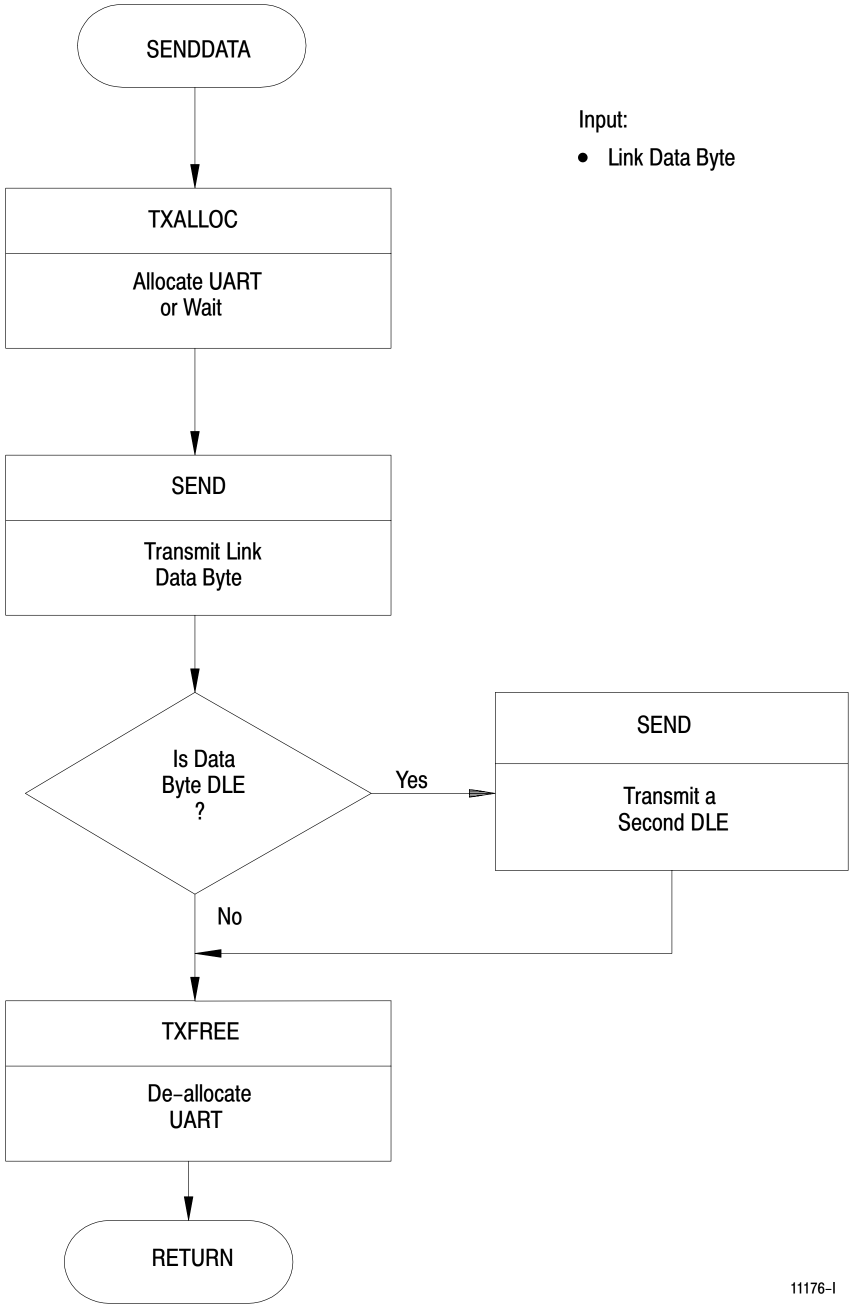
- TXWAIT: A Sleep Variable


SEND
Disable Processor Interrupts
Enable UART Transmit Interrupt
UART Transmitter Empty ?
No
SLEEP
Suspend at TXWAIT
Yes
Disable UART Transmit Interrupt
Output Byte to UART
Enable Processor Interrupts
Indivisible Zone
Return

NOTE: This figure assumes the use of a Z80 SIO

11175–I

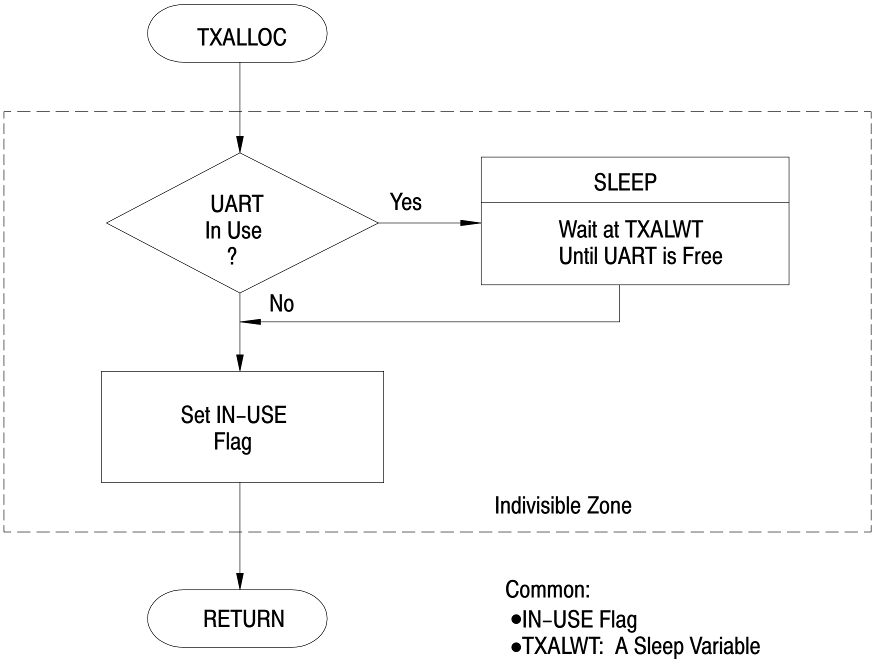
Figure B.14
SENDDATA Subroutine



Input:

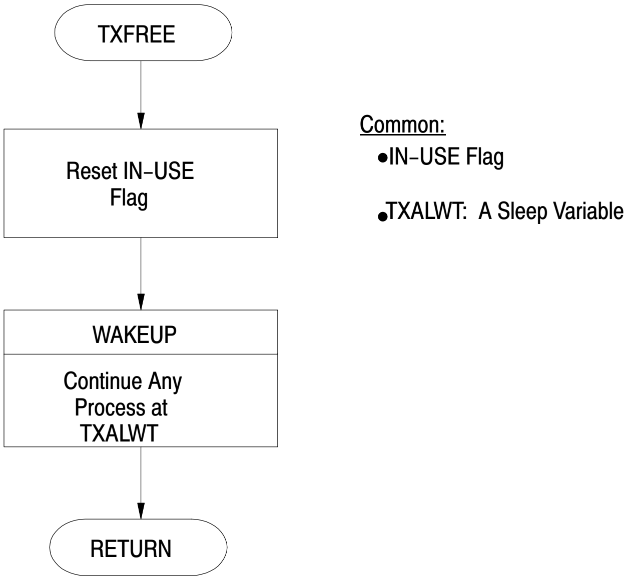
- Link Data Byte

Figure B.15
TXALLOC Subroutine



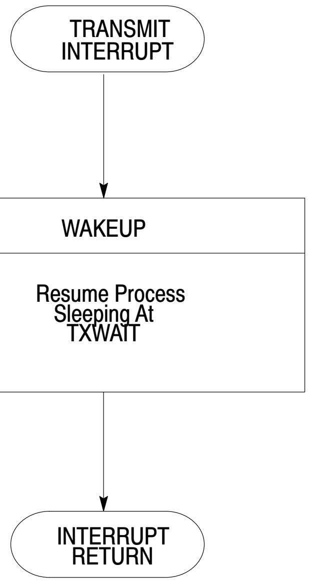
11177–I

Figure B.16
TXFREE Subroutine



11178–I

Figure B.17
TRANSMIT INTERRUPT Subroutine

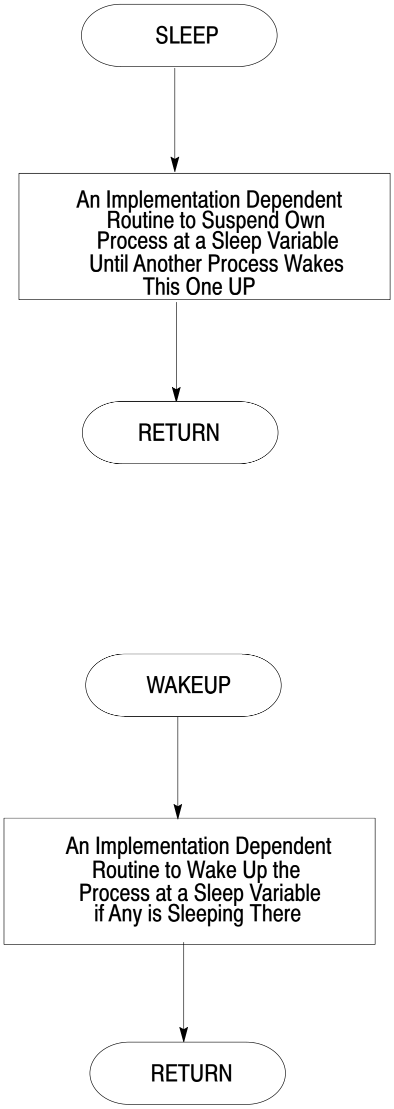


NOTE: This figure assumes the use of a Z80 S10.

NOTE: UART transmit interrupt must be enabled and disabled without affecting the current state of the **receive** and **status interrupt** enable flags.

11179–I

Figure B.18
SLEEP AND WAKEUP Subroutines


SLEEP
An Implementation Dependent Routine to Suspend Own Process at a Sleep Variable Until Another Process Wakes This One UP
RETURN
WAKEUP
An Implementation Dependent Routine to Wake Up the Process at a Sleep Variable if Any is Sleeping There
RETURN

Input:

- The Address of a Sleep Variable.

Sleep Variables:

- Typically an address of a stack or a process or context save area.
- A Process can suspend itself and place its address in a sleep variable.
- Subsequently another process can wake up the sleeping process by referring to sleep variable. When no process is sleeping at a sleep variable, a WAKEUP has no effect.

Input:

- The Address of a Sleep Variable.

11180–I

Figure B.19
SLEEP and WAKEUP Interaction

NOTE: SLEEP and WAKEUP are always used in connection with some type of indivisible interprocess interlock. Indivisibly is achieved on many processors (e.g. – Z–80) by disabling processor interrupts. For this reason SLEEP and WAKEUP assume that interrupts are off when they are called. They will always return with interrupts off.

The interaction of SLEEP and WAKEUP:

When one process calls SLEEP, the result is a return from a WAKEUP by another process. When a process calls WAKEUP, the result is a return from a call to SLEEP by another process. An interrupt subroutine that calls WAKEUP is viewed as a subroutine of the interrupted process.

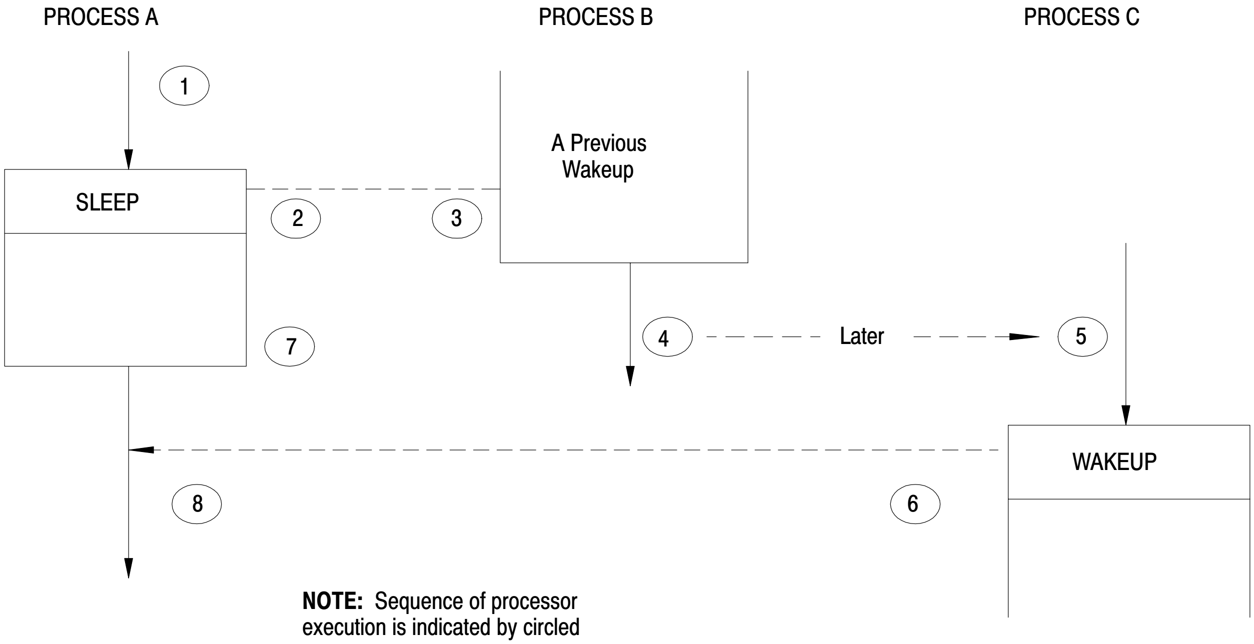
In the example in Figure B–18 Process B woke up Process A some time ago. Now, at 1, when A goes to sleep, actual execution resumes after the wakeup call in B at 3 and 4. Sometime later, process C (at an interrupt, for example) calls WAKEUP at 5. Execution flow proceeds to the instructions at 8 following the call to SLEEP in Process A. The next time A calls SLEEP, he WAKEUP call in C will terminate.

NOTE: This is not the only possible implementation of SLEEP and WAKEUP.

Another implementation would allow a process to call WAKEUP without losing immediate control of the processor. Context switching would be deferred until B itself executed a SLEEP.

A third alternative would cause a context switch of a process performed a WAKEUP on a higher priority process. If a WAKEUP was performed on a lower priority process the context switch would be deferred until the first process went to SLEEP.

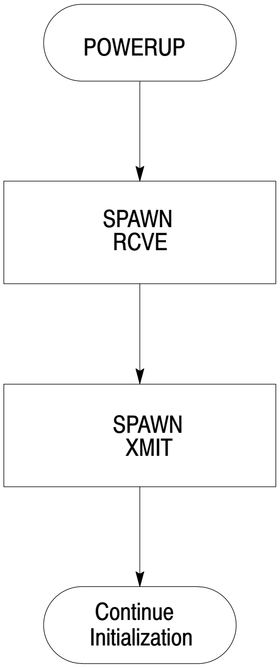
The first alternative is the result of implementing the driver totally at interrupt level. The third alternative would be used if the driver were implemented as tasks – under a multitasking operating system. Such an implementation might be easier, but would probably be limited to lowever baud rates.



NOTE: Sequence of processor execution is indicated by circled numbers.

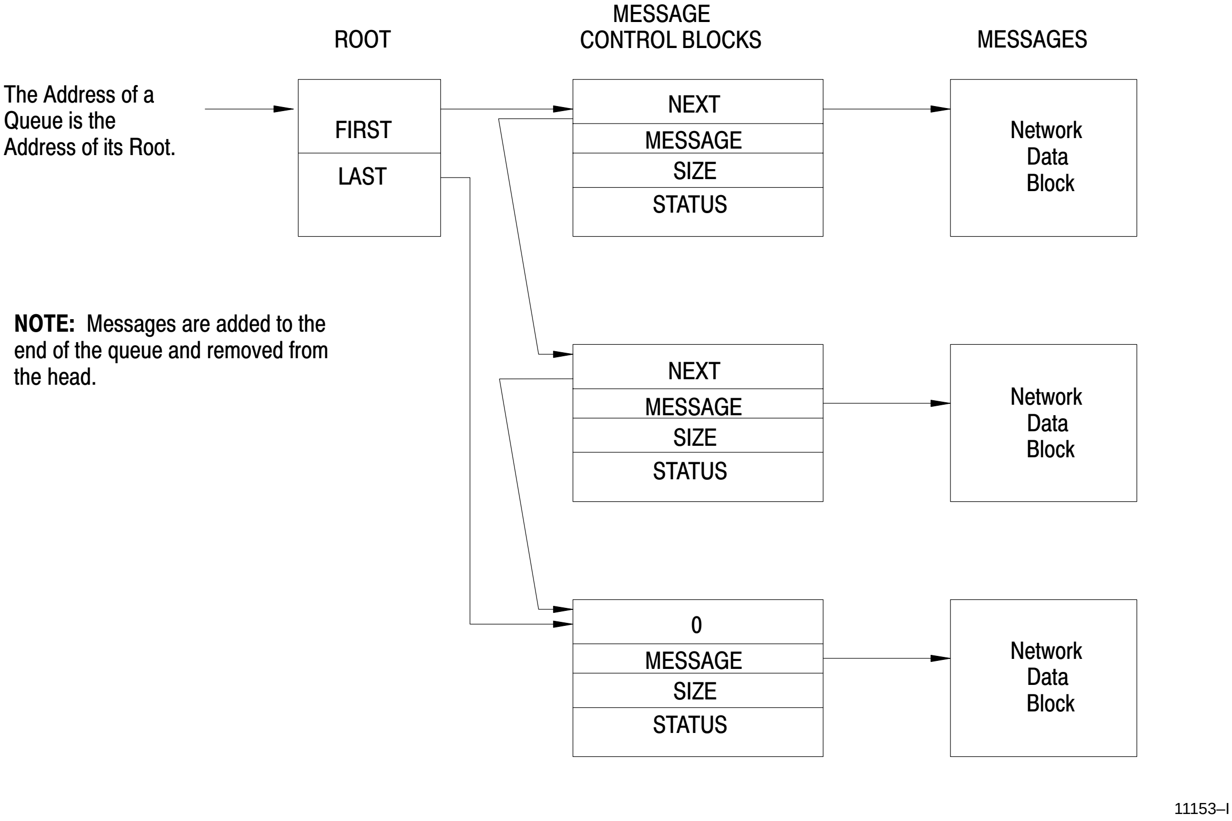
Figure B.20
POWERUP Routine

Powerup: At powerup the Z-80 starts executing code at location 0. The powerup routine starts the XMIT and RCVE processes by executing a SPAWN. A SPAWN is very similar to a WAKEUP, except the corresponding SLEEP is imaginary, and is located ahead of the first instruction of the SPAWNed process.



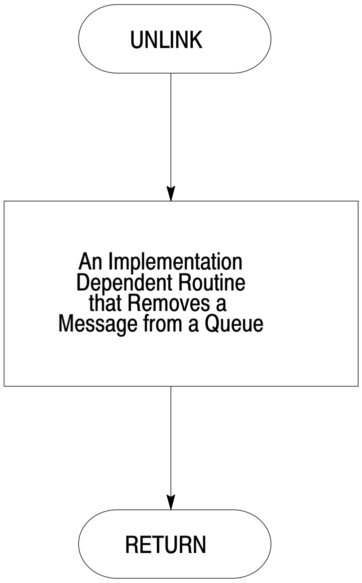
11182–I

Figure B.21
Message Queue



NOTE: Messages are added to the end of the queue and removed from the head.

Figure B.22
UNLINK Subroutine


UNLINK
An Implementation Dependent Routine that Removes a Message from a Queue
RETURN

Input:

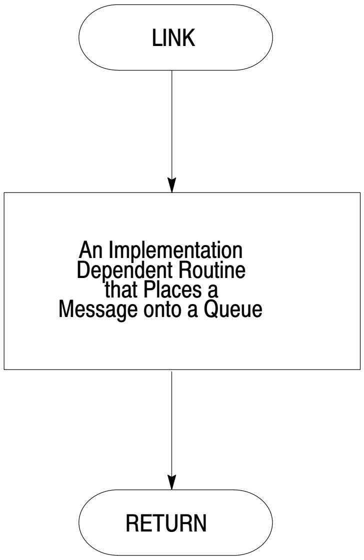
- Address of Queue

Output:

- Message Control Block

11184–I

Figure B.23
LINK Subroutine

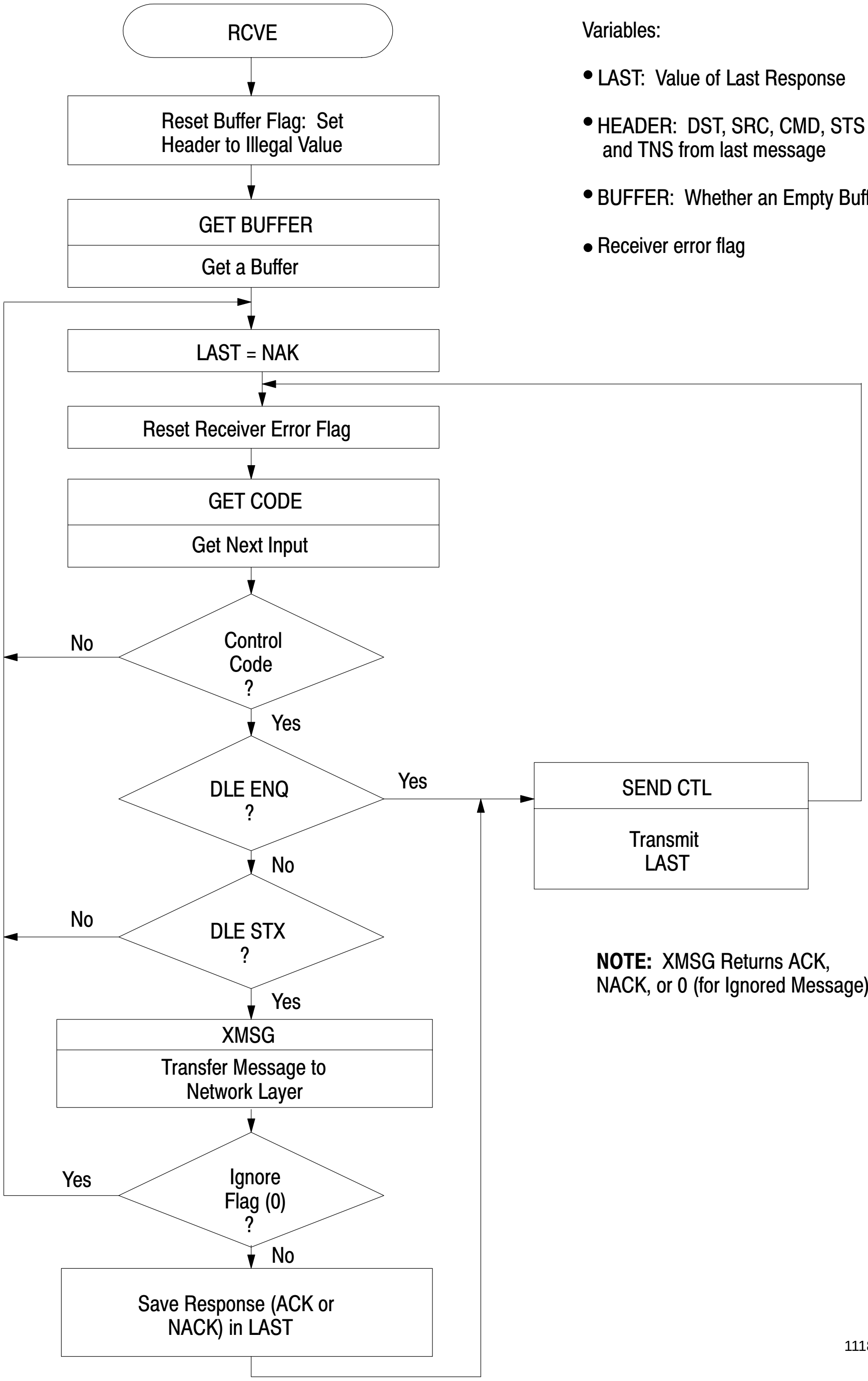

LINK
An Implementation Dependent Routine that Places a Message onto a Queue
RETURN

Input:

- Address of Queue Message Block

11185–I

Figure B.24
Receiver Routine for Full-Duplex Protocol



Variables:

- LAST: Value of Last Response
- HEADER: DST, SRC, CMD, STS and TNS from last message
- BUFFER: Whether an Empty Buffer is Read
- Receiver error flag

NOTE: XMSG Returns ACK, NACK, or 0 (for Ignored Message)

11186–I

Figure B.25
XMSG Subroutine

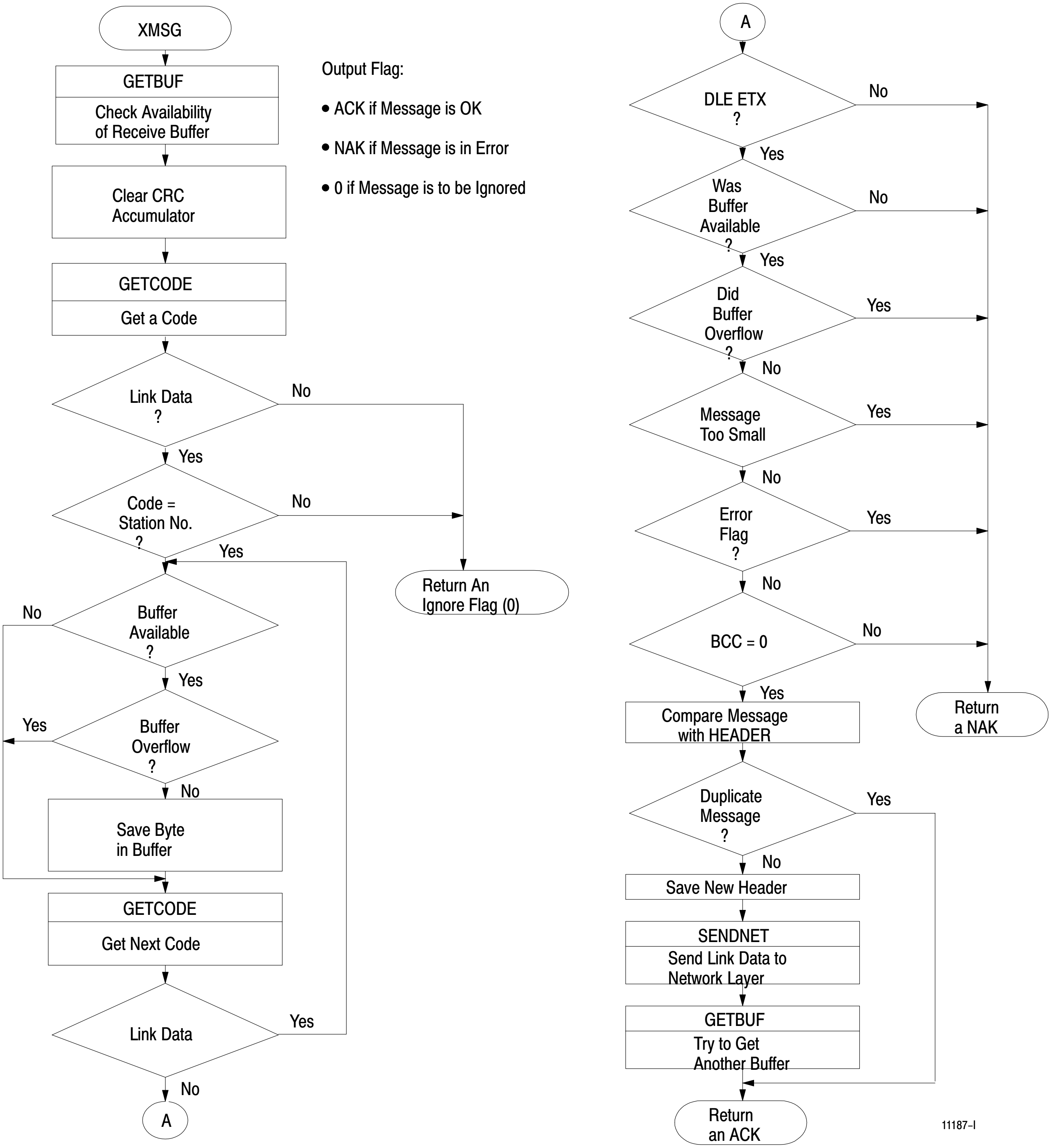
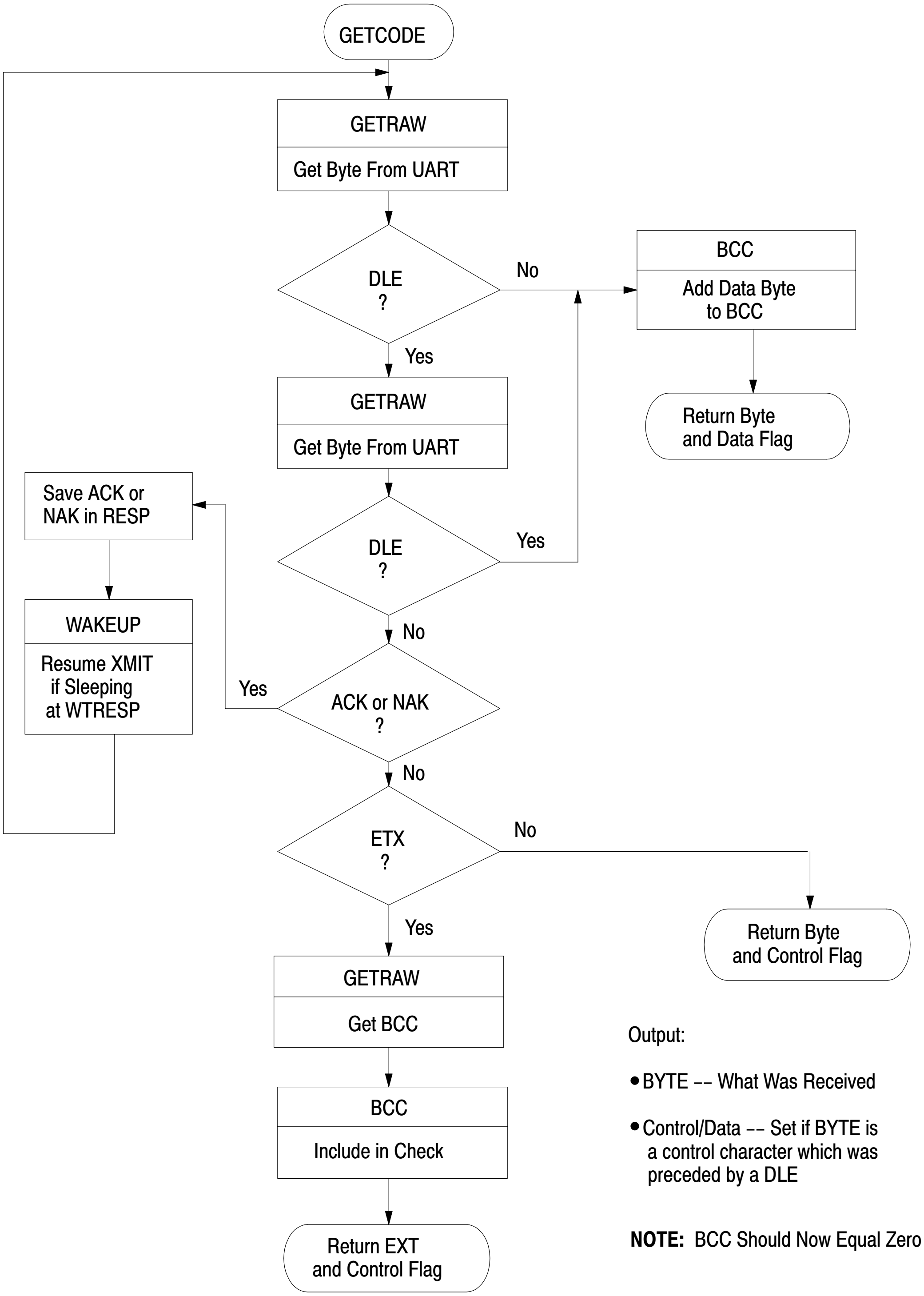


Figure B.26
GETCODE Subroutine



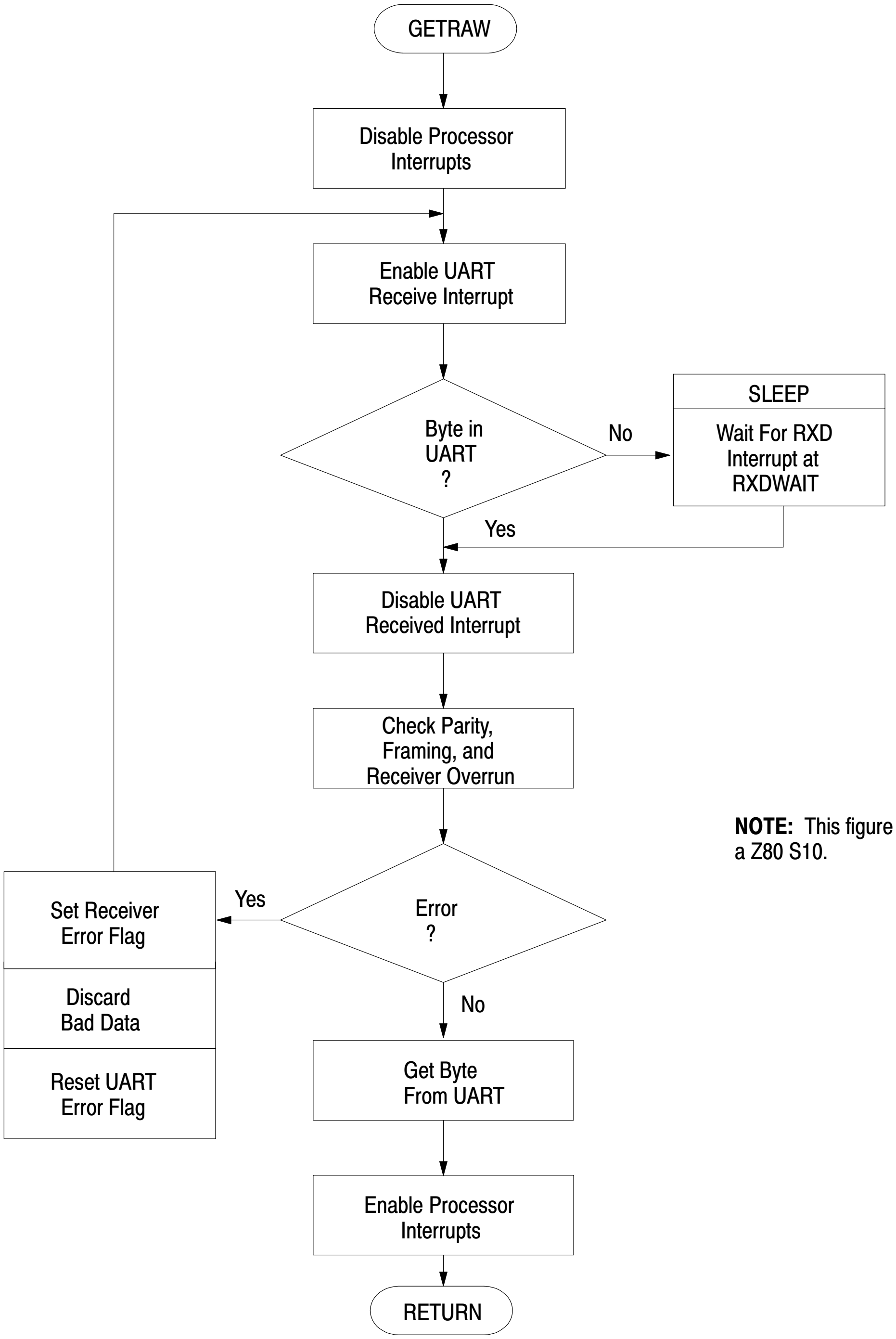
Output:

- BYTE -- What Was Received
- Control/Data -- Set if BYTE is a control character which was preceded by a DLE

NOTE: BCC Should Now Equal Zero

11188–I

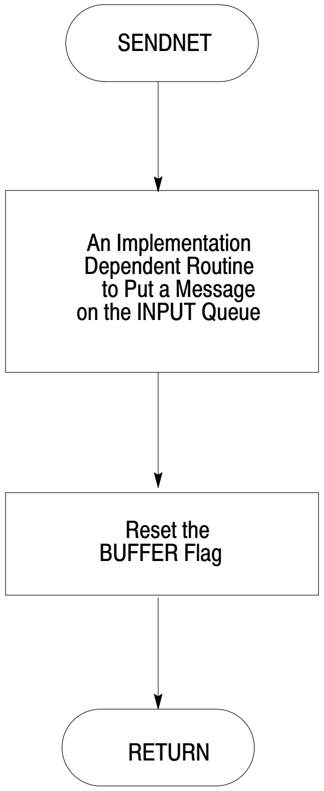
Figure B.27
GETRAW Subroutine


GETRAW
Disable Processor Interrupts
Enable UART Receive Interrupt
Byte in UART ?
No
SLEEP
Wait For RXD Interrupt at RXDWAIT
Yes
Disable UART Received Interrupt
Check Parity, Framing, and Receiver Overrun
Error ?
Yes
Set Receiver Error Flag
Discard Bad Data
Reset UART Error Flag
No
Get Byte From UART
Enable Processor Interrupts
RETURN

NOTE: This figure assumes the use of a Z80 S10.

11189–I

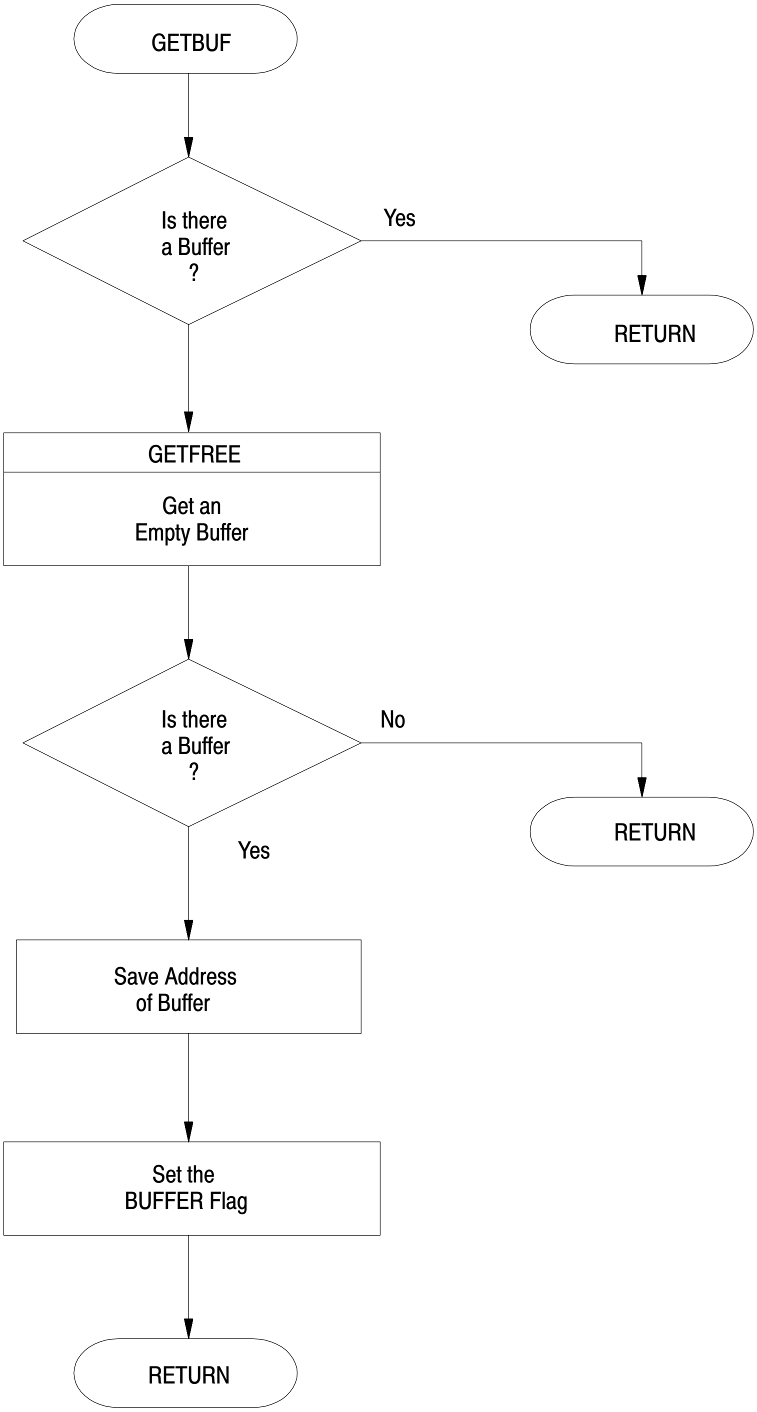
Figure B.28
SENDNET Subroutine


SENDNET
An Implementation Dependent Routine to Put a Message on the INPUT Queue
Reset the BUFFER Flag
RETURN

Input:

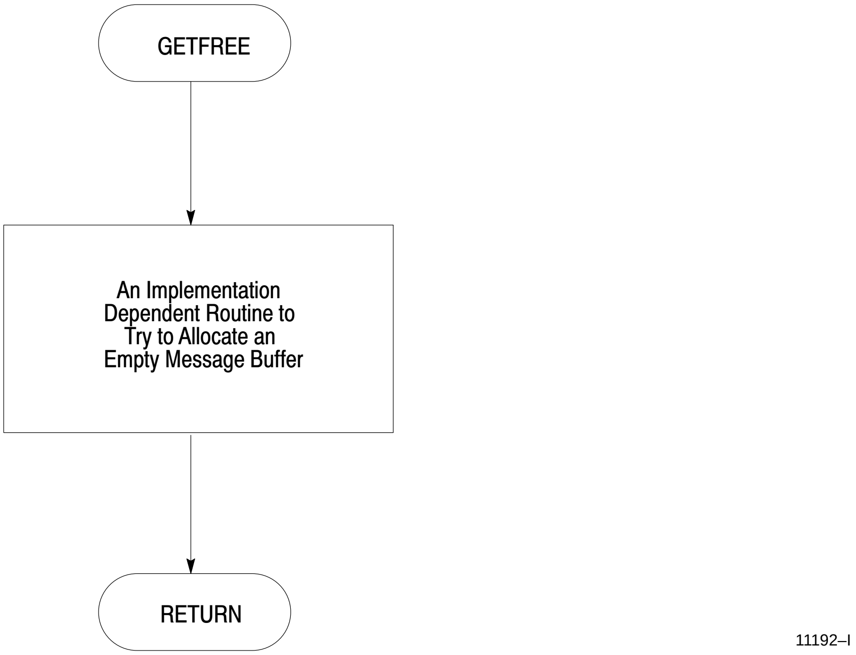
- Message Buffer

Figure B.29
GETBUF Subroutine



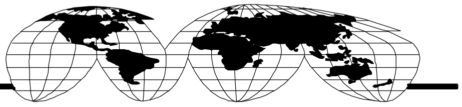
11191–I

**Figure B.30
GETFREE Subroutine**


GETFREE
An Implementation Dependent Routine to Try to Allocate an Empty Message Buffer
RETURN
11192–I

Rockwell Automation
Allen-Bradley

Allen-Bradley, a Rockwell Automation Business, has been helping its customers improve productivity and quality for more than 90 years. We design, manufacture and support a broad range of automation products worldwide. They include logic processors, power and motion control devices, operator interfaces, sensors and a variety of software. Rockwell is one of the worlds leading technology companies.



Worldwide representation.

Argentina • Australia • Austria • Bahrain • Belgium • Brazil • Bulgaria • Canada • Chile • China, PRC • Colombia • Costa Rica • Croatia • Cyprus • Czech Republic • Denmark • Ecuador • Egypt • El Salvador • Finland • France • Germany • Greece • Guatemala • Honduras • Hong Kong • Hungary • Iceland • India • Indonesia • Ireland • Israel • Italy • Jamaica • Japan • Jordan • Korea • Kuwait • Lebanon • Malaysia • Mexico • Netherlands • New Zealand • Norway • Pakistan • Peru • Philippines • Poland • Portugal • Puerto Rico • Qatar • Romania • Russia–CIS • Saudi Arabia • Singapore • Slovakia • Slovenia • South Africa, Republic • Spain • Sweden • Switzerland • Taiwan • Thailand • Turkey • United Arab Emirates • United Kingdom • United States • Uruguay • Venezuela • Yugoslavia

Allen-Bradley Headquarters, 1201 South Second Street, Milwaukee, WI 53204 USA, Tel: (1) 414 382-2000 Fax: (1) 414 382-4444

Publication 1771–6.5.15 – February, 1985
Supersedes Publication 1771–822– March, 1984

PN 955096–87